

# ReactJS Redux, the Right Way.

## This is the **React Convention™**: convention over configuration step-by-step guide for the React Native and ReactJS developers.

This is the online convention and tutorial's e-book for [ReactJS](#) Developers. Learn the React Redux best practises.

**This e-book is 100% online** (no emails required, no PDFs to download etc.). It contains learning materials for junior and senior front-end devs who want to master React Redux ([for React Native tutorial click here](#)).

In case of any problems and feedback, feel free to contact me at [kamil.przeorski@gmail.com](mailto:kamil.przeorski@gmail.com)

## The React Convention™ Tutorial's Content Table

### Chapter #1: ReactJS

- Why to learn and use React
- Why Redux
- Learning ES6
- How to learn basic things in React and Redux
- Why to use the React Convention™
- How to use the React Convention™ e-book
- Your first step with the React Convention™
- General client codebase structure of the React Convention™
- New dashboard route
- Create a dashboard component (from the Counter copy)
- Next steps in implementing our dashboard
- Mocked items list on the dashboard list
- Add/edit item on the dashboard list
- Reorder an item on the dashboard list (pure React Drag and Drop example)
- Login with mocked data (front-end)
- A summary (front-end part)

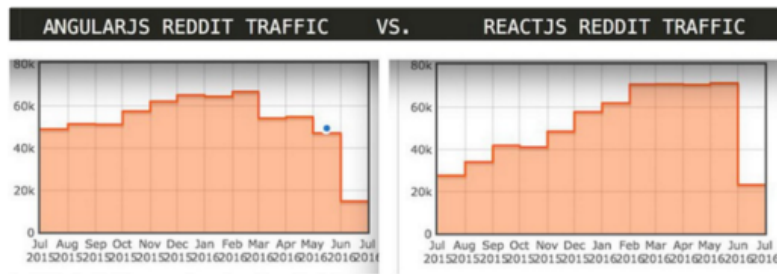
### Chapter #2: React Native

For the React Native content table please [click here](#).

## Why you should learn and use React

### 1) Because of timing

Currently, React has the best momentum between all the front-end frameworks/solutions/libraries. Below we can find the stats of visits within Angular vs. React subreddits:



(reddit's stats from July 2016)

Above, we can see that React subreddit has overpasses with popularity (unique visits) the AngularJS.

"A good hockey player plays where the puck is. A great hockey player plays where the puck is going to be" - Wayne Gretzky

As you can see, **the advice from Wayne Gretzky may be quite useful: play where "the puck" is going to be (React, not Angular).**

ReactJS related skills are going to be very demanded on the job market. [Here you can learn basics and advanced concepts behind React Redux, the right way.](#)

## 2) Because of better job market

Who Is Hiring on Hacker News are monthly updates of tech companies who are looking for certain skillsets:

On The Hacker News, the React popularity in the WhoIsHiring has passed the Angular's one. You can find it on the trend graph below:



The black line is Angular.

The light blue line is React.

The screenshot above shows the stats of Angular and React. Where the black line that's down-trending represents Angular and the light-blue line that's up-trending represents React.

**Second part of that point is the geography.**

Angular is searched mostly in poor cities while React has more traction in SF, Sydney or Austin:

**Google Trends for Angular.js (July 2016):****Google Trends for React.js (July 2016 - still not perfect, but looks better than Angular):**

As you can see on the locations' list - the Angular.JS is mostly searched in countries like India and the React.JS is more popular in places creating **TRENDS** as for example San Francisco, Sydney or Austin (that mean, it's going to be more popular over time).

**3) Because of it's simplicity over Angular and other**

React.JS API is short and sweet. Redux helps in building more complex apps. You can learn more about both below.

**4) Because of it's robust ecosystem**

React has the most robust community in terms of modular components which are shared online for reusing purpose.

## Why Redux

Redux is "kind of" replacement of Service or Factory in Angular. It keeps your application model in one place and helps organize data of your application.

**Redux's ANALOGY #1:** Redux is "kind of back-end on the front-end" - it's your database of your front-end application.

**Redux's ANALOGY #2 (TO SQL):** it helps you to query SELECT, INSERT and UPDATE the client-side data in it's single state tree (you will learn about it later).

In that e-book you will learn the basics and advanced things regarding "how to use React Redux, the right way".

**Why to use Redux:****1) It's simpler to learn (than Facebook's FLUX singleton approach)**

From my experience of teaching developers FLUX and Redux - I have found that Redux is easier to grasp.

**2) It's more powerful than other solutions**

It has many features which are unique and very useful in real life web apps development (like immutability).

**3) It's the most used across the React community**

Redux is the most standard tool to use for the uni-directional data flow.

A good example is the [ReactJS Convention™](#) which you can learn here - it deploys Redux in 100%.

#### 4) Server side rendering with Redux is great

Redux is one of best solutions for server-side React's applications. It's very unique and useful feature that is mostly used on websites where the user experience must be top-notch (the less latency of the server's response the better) - for example Airbnb or most popular news services have server rendering implemented, so that their users can have content served much quicker than in a regular (non server-side rendering) single-page-application.

## Learning EcmaScript6 (JS)

---

I assume that most people who came to the [ReactJS Convention™](#) e-book's website don't know too much about React and Redux.

**If you feel you are strong in using React and Redux then you can skip Preface #1 and go directly to [Why you should use the React Convention™](#) and start learning advanced things.**

### **Knowledge about React and Redux is very important to understand concepts behind React Convention™.**

Because of that I will share with you online resources which are the most useful for a someone who:

- Has background in HTML5, CSS3 and VanillaJS
- Likely, has experience in jQuery, Angular or other "older" front-end solutions
- Wants to learn ReactJS basics
- Likes the React Convention™ idea and wants to make highly scalable and manageable client-side apps

### **THINGS TO LEARN AND MASTER before advancing into the React Convention™ concept:**

#### **1) Functional programming concepts in VanillaJS:**

##### **Higher-order functions - Part 1 of Functional Programming in JavaScript :**

<https://www.youtube.com/watch?v=BMUiFMZr7vk>

##### **Map - Part 2 of Functional Programming in JavaScript :**

<https://www.youtube.com/watch?v=bCqtb-Z5YGQ>

##### **Reduce basics - Part 3 of Functional Programming in JavaScript :**

<https://www.youtube.com/watch?v=Wl98eZpkp-c>

##### **Reduce Advanced - Part 4 of Functional Programming in JavaScript:**

<https://www.youtube.com/watch?v=1DMolJ2FrNY>

##### **Closures - Part 5 of Functional Programming in JavaScript :**

<https://www.youtube.com/watch?v=CQqwU2Ixu-U>

##### **Currying - Part 6 of Functional Programming in JavaScript :**

<https://www.youtube.com/watch?v=iZLP4qOwY8I>

**Recursion - Part 7 of Functional Programming in JavaScript :**

<https://www.youtube.com/watch?v=k7-N8R0-KY4>

**Promises - Part 8 of Functional Programming in JavaScript:**

<https://www.youtube.com/watch?v=2d7s3spWAzo>

If you are a JS developer, then you should be already familiar with the concepts from the links above (^ ^). I'm putting the links for other people who want to master the JS. **In React we use much more VanillaJS than in other solutions like AngularJS or EmberJS.**

## How to learn React and Redux basics

---

After you are familiar with the tutorials linked to YouTube from the above list, then you can start learning ES6 and more JS from Egghead (or skip it and go directly to resources related to React and Redux in case if you feel strong in that area).

### 2) Egghead resources about ES6 and JS:

**EcmaScript 6:**

<https://egghead.io/courses/learn-es6-ecmascript-2015>

**Reduce Data with Javascript Array:**

<https://egghead.io/courses/reduce-data-with-javascript>

**JavaScript Arrays in Depth:**

<https://egghead.io/courses/javascript-arrays-in-depth>

### 3) Useful resources about React and Redux:

**React Fundamentals:**

<https://egghead.io/courses/react-fundamentals>

**Important Redux's Three Principles:**

<https://github.com/reactjs/redux/blob/master/docs/introduction/ThreePrinciples.md>

You need to be aware of three principles:

- Single source of truth
- State is read-only
- Changes are made with pure functions

**All the three principles are explained in that video series: "Getting Started with Redux":**

<https://egghead.io/courses/getting-started-with-redux>

Once you are familiar with the materials listed above, then you are ready to learn the React Convention™ in order to make SPA apps, the right way.

## Why you should use the React Convention™

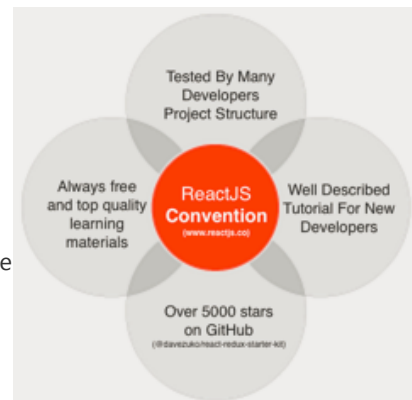
"So many projects, so little time." - Kamil Przeorski

### ReactJS Convention™ connects different things.

Learning materials + tested project structure with over 5000 stars + support on [FB group](#) in case any problems.

This is the best place in the internet, where you can find all the learning materials about **React Redux, the right way**. You can be sure that after finishing this guide you will be able to make client-side single-page-apps which are profesional and scalable.

Have fun while learning :-)



In 2013 ReactJS was almost "nothing", today it's "something" for the JavaScript world. We developers, have so little time and so much work to do. The main problem with anyone who wants to learn how to make single-page-apps with React is that there **are no really conventions that one can follow**. I am writing about those problems as a founder of the ReactJS/React-Native Webshop called MobileWeb Pro ([www.MWP.io](http://www.MWP.io)) - we made over 15+ React projects for clients from 4 continents **and I can tell you that we faced so many frustrations (and this is why we want to share with you the ReactJS Convention™ for free)**. Mostly when a client who had already an MVP wanted us to expand based on his codebase. **No convention at all, many projects structure codebases and almost each one had it's weak points - this led us to many frustrations**. Also, another source of annoyance is, when you need to introduce a new developer to your "well known to you" codebase **and you want to make sure that he doesn't make any "newbie" mistakes**. In context of this preface a "newbie" means a someone who isn't acquainted with the project's structure. Because he is not familizarized with the project's configuration, then he is not really productive in his first month of work and additionally slows your progress down. **You need to teach him about your configuration which will take you decent amount of time to pass all your knowledge to your new teammate in the project**. The React Convention™ e-book helps you to save time on transferring knowledge about your project's structure to any new member.

The main ReactC's goal is to create "ready to go" convention that you can share with your colleagues.

### SHARE THE REACT CONVENTION IF YOU LIKE THE IDEA.

We will familiarize your new teammate with your React's project structure so that you will save a lot of money when following it. You will save 90% of your time on explaining why, how and what has been done with the project's codebase.

**The ReactC e-book is also a good for people who want to start a new project from scratch as it guides you on how to properly equip the new starter with your codebase and guarantees that any new future members will be introduced to your codebase quicker than any other method.**

**Welcome to the React Convention™ way of doing single-page apps.**

**PLEASE SEND US FEEDBACK, SO WE CAN IMPROVE THAT BOOK BASED ON YOUR FINDINGS AND ISSUES - mail us with details at [kamil.przeorski@gmail.com](mailto:kamil.przeorski@gmail.com)**

Like

Share

103 people like this. Be the first of your friends.

## How to use the React Convention™ e-book

---

In order to get benefits, you need already to be familiarized with:

- 1) ES6/ES7 syntax
- 2) Concept of React Life Cycle
- 3) Basic Redux stuff

All the above things can be learned from the [How to learn basic things in React and Redux](#). If you are already familiar with all that basic knowledge, then please continue.

**IMPORTANT:** if you are a beginner and you need a list of resources where to learn those three things, then join the FB group [ReactJS Convention™ Facebook Group](#). I will share it with you there. Join the [group](#) if you have any trouble with this e-book as well.

Other requirement required to get all the benefits mentioned in the preface is that you will need to start your project with that react-redux-starter-kit:

<https://github.com/davezuko/react-redux-starter-kit>

We have found this starter, best (we know what we are talking about because we made over 15+ different React projects so far). Even Dan Abramov has noticed that this starter kit is really an awesome one:

Dan Abramov is mostly famous for creating Redux and currently he is the main guy behind the ReactJS development team (he works for Facebook).



**Dan Abramov**  
@dan\_abramov



Following

Terrific universal Redux + React Router starter kit. Love it. [github.com/davezuko/react ...](https://github.com/davezuko/react-redux-starter-kit)

[www.reactjs.co/](http://reactjs.co/)

Once you start using this starter kit, definitely the e-book will provide you the value which will save your team a plenty of time at work.

In case of any issue, we are on Facebook in order to help you - just ping us on [the FB](#) or send an email.

## Your first step with the React Convention™

---

In a default starter kit from the @davezuko, you can see a home page and a counter component which look exactly as the image below:



## React Redux Starter Kit

Home · [Counter](#)

Counter: **16**

Increment

Double (Async)

This is a simple application with a counter, and through this e-book you will implement other views and components which will help you understand whole codebase from a big picture perspective.

To explain the whole code structure, we will start with developing a new route called Dashboard with some basic features. When we are done with it, then we will implement a registration and login which will ensure that you know how to do it 100% just on your own.

First step is to clone the starter kit and start working on this commit (the e-book was written on that commit, so for better experience you shall work on the same one):

```
9a03e99c0dd2e7102d43264cc495bbdd4e10dcdd
```

... so you will be truly working on that starting codebase:

```
https://github.com/davezuko/react-redux-starter-kit/tree/9a03e99c0dd2e7102d43264cc495bbdd4e10dcdd
```

After you have cloned the repo and you are at the same commit as me, then we can continue with the fun.

## General client codebase structure of the React Convention™

We will focus on the client-side explanation for now, so let's discuss the "src" directory that has following structure:

```

|— src                                # Application source code
|   |— index.html                    # Main HTML page container for app
|   |— main.js                       # Application bootstrap and rendering
|   |— components                    # Reusable Presentational Components
|   |— containers                    # Reusable Container Components
|   |— layouts                       # Components that dictate major page structure
|   |   |— CoreLayout.js             # CoreLayout which receives children for each route
|   |   |— CoreLayout.scss           # Styles related to the CoreLayout
|   |   |— index.js                  # Main file for layout
|   |— routes                        # Main route definitions and async split points
|   |   |— index.js                  # Bootstrap main application routes with store
|   |   |— Home                      # Fractal route
|   |   |   |— index.js              # Route definitions and async split points
|   |   |   |— assets                # Assets required to render components
|   |   |   |— components            # Presentational React Components
|   |   |   |— routes **             # Fractal sub-routes (** optional)
|   |   |— Counter                  # Fractal route
|   |       |— index.js              # Counter route definition
|   |       |— container             # Connect components to actions and store
|   |       |— modules               # Collections of reducers/constants/actions

```



```

├── routes **      # Fractal sub-routes (** optional)
├── static         # Static assets (not imported anywhere in source code)
├── store         # Redux-specific pieces
├── createStore.js # Create and instrument redux store
├── reducers.js   # Reducer registry and injection
├── styles        # Application-wide styles (generally settings)
└── tests         # Unit tests

```

If you were working on any FLUX or redux projects, then the project structure should be quite familiar to you. You will learn all the things located in that project step-by-step (bottom-up approach).

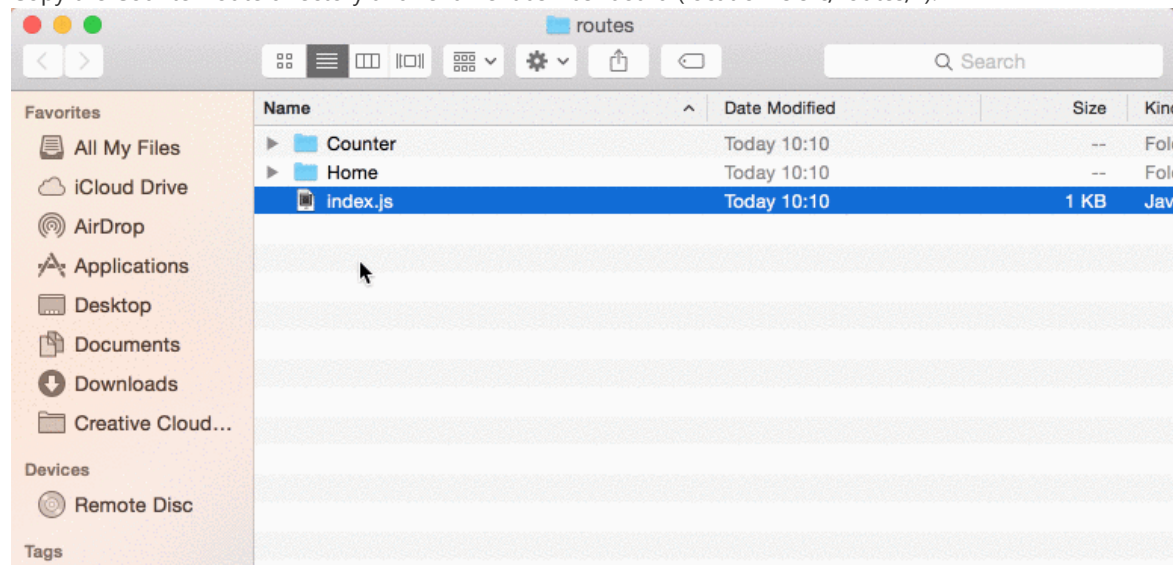
Don't worry if you don't get the structure so far, you will ramp up with knowledge about it during the following instructions. We will also explain each step so you will learn by doing.

Let's start with implementing a new route called "DASHBOARD".

Have you cloned the repo to your local machine? Great, now you can follow the steps below.

## New dashboard route

Copy the Counter route directory and rename it as Dashboard (location is src/routes/\*):



REACTJS    REACT NATIVE    BLOG    COMMUNITY

Yellow

**IMPORTANT:** from that point we will create a new component based on a copy of the Counter route in the `src/routes/Counter` in the `@davezuko's redux starter`.

Next step is to find any related things to counter and then:

- rename things from counter to dashboard
- We will rename action called **COUNTER\_INCREMENT** to **DASHBOARD\_VISITS\_COUNT** during one sessions without refreshing the browser)

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

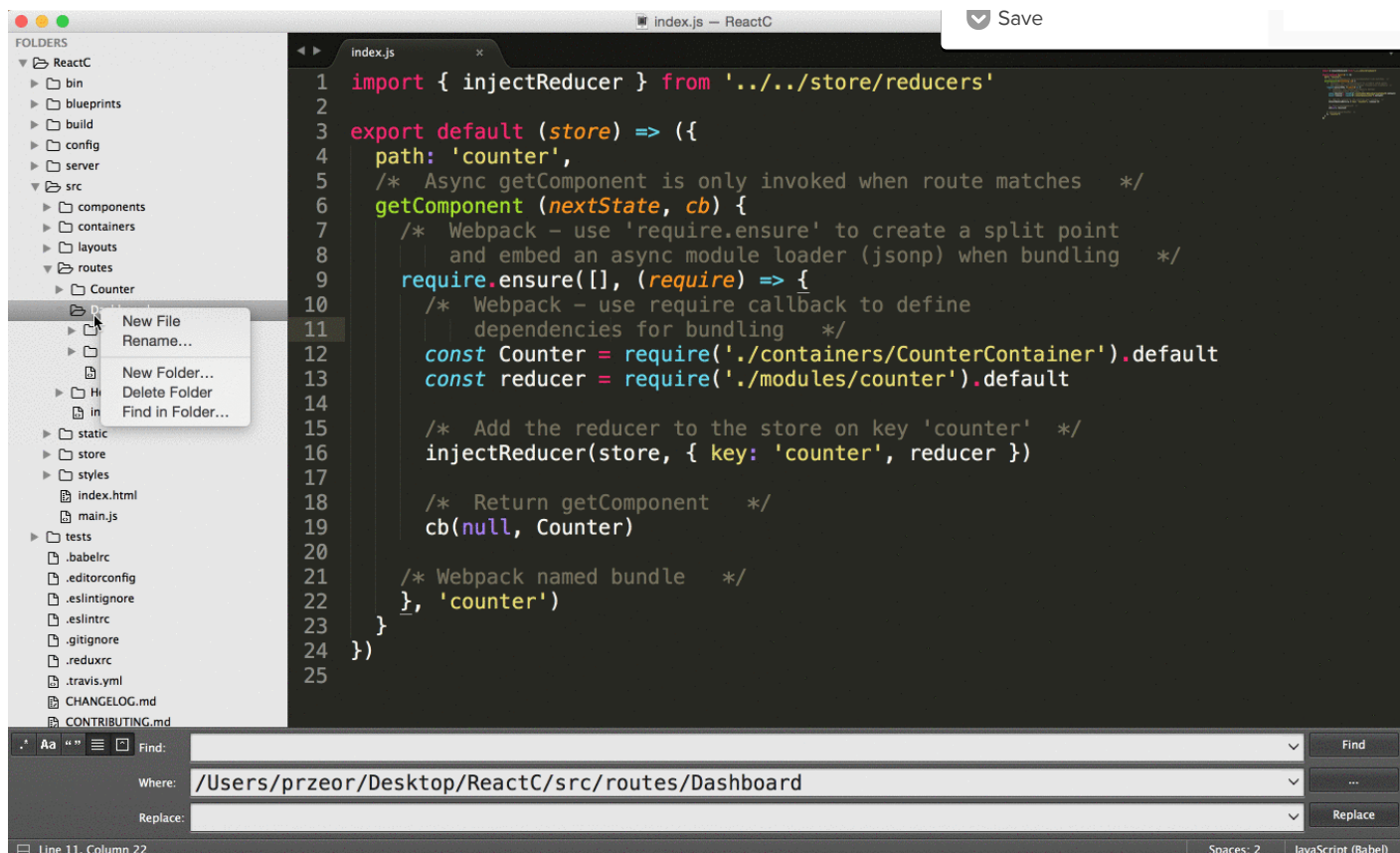
Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips SPONSORED  
Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com



Generally all the code diffs below are simply copies of Counter's component renamed with Dashboard.

Source code of the new dashboard init creation: <https://github.com/przeor/ReactC/commit/d3f5d0293045af4ce75522324c06c9bf44>

## Create a dashboard component (from the Counter copy)

We have to copy the directory from src/components/Counter and name it Dashboard. Then rename all the variables, values and comments related to counter's route as on the below's example:

REACTJS | REACT NATIVE | BLOG | COMMUNITY

src/components/Dashboard/Dashboard.js

**Tags Saved** ...

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

### Our First 50,000 Stars

facebook.github.io

Save

### From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

### Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

29 src/components/Dashboard/Dashboard.js Save

... @@ -0,0 +1,29 @@

1 +import React from 'react'  
2 +import classes from './Dashboard.scss'  
3 +  
4 +export const Dashboard = (props) => {  
5 + <div>  
6 + <h2 className={classes.dashboardContainer}>  
7 + Dashboard:  
8 + {' '  
9 + <span className={classes['dashboard--green']}>  
10 + {props.dashboard}  
11 + </span>  
12 + </h2>  
13 + <button className='btn btn-default' onClick={props.increment}>  
14 + Increment  
15 + </button>  
16 + {' '  
17 + <button className='btn btn-default' onClick={props.doubleAsync}>  
18 + Double {Async}  
19 + </button>  
20 + </div>  
21 +}  
22 +  
23 +Dashboard.propTypes = {  
24 + dashboard: React.PropTypes.number.isRequired,  
25 + doubleAsync: React.PropTypes.func.isRequired,  
26 + increment: React.PropTypes.func.isRequired  
27 +}  
28 +  
29 +export default Dashboard

www.reactjs.co

FYI: For better code diffs quality, please click on it.

New file (copied from Counter example - you can click the diffs image to make it larger):  
src/components/Dashboard/Dashboard.scss

12 src/components/Dashboard/Dashboard.scss View

... @@ -0,0 +1,12 @@

1 +.dashboard {  
2 + font-weight: bold;  
3 +}  
4 +  
5 +.dashboard--green {  
6 + composed: dashboard;  
7 + color: rgb(25,200,25);  
8 +}  
9 +  
10 +.dashboardContainer {  
11 + margin: 1em auto;  
12 +}

www.reactjs.co

REACTJS

REACT NATIVE

BLOG

COMMUNITY

New file (copied from Counter example - you can click the diffs image to make it  
src/components/Dashboard/index.js

3 src/components/Dashboard/index.js

... @@ -0,0 +1,3 @@

1 +import Dashboard from './  
2 +  
3 +export default Dashboard

www.reactjs.co

FYI: For better code diffs quality, please click on it.

Next step is to create a link in the Header component:

Modfiy the file (you can click the diffs image to make it larger):  
src/components/Header/Header.js

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips SPONSORED

Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com

4

src/components/Header/Header.js

Save

12

<Link to="/counter" activeClassName={classes.activeRoute}>

13

Counter

14

</Link>

15

</div>

16

}

17

12

<Link to="/counter" activeClassName={classes.activeRoute}>

13

Counter

14

</Link>

15

+ { ' - ' }

16

+ <Link to="/dashboard" activeClassName={classes.activeRoute}>

17

+ Dashboard

18

+ </Link>

19

</div>

20

}

21

www.reactjs.co

FYI: For better code diffs quality, please click on it.

Further, we simply need to rename the files and replace all the "counter" matches to "dashboard" (again):

Renaming and changes in (you can click the diffs image to make it larger):  
.../Dashboard/containers/CounterContainer.js → ...ashboard/containers/DashboardContainer.js

16

.../Dashboard/containers/CounterContainer.js → ...ashboard/containers/DashboardContainer.js

View

1

import { connect } from 'react-redux'

2

-import { increment, doubleAsync } from '../modules/counter'

3

4

/\* This is a container component. Notice it does not contain any JSX,

5

nor does it import React. This component is «only» responsible for

6

wiring in the actions and state necessary to render a presentational

7

- component - in this case, the counter: \*/

8

9

-import Counter from 'components/Counter'

10

11

/\* Object of action creators (can also be function that returns object).

12

Keys will be passed as props to presentational components. Here we are

13

14

@@ -10,21 +10,21 @@ const mapActionCreators = {

15

16

}

17

18

const mapStateToProps = (state) => ({

19

- counter: state.counter

20

})

21

22

/\* Note: mapStateToProps is where you should use 'reselect' to create selectors, ie:

23

24

import { createSelector } from 'reselect'

25

- const counter = (state) => state.counter

26

- const tripleCount = createSelector(counter, (count) => count \* 3)

27

const mapStateToProps = (state) => ({

28

- counter: tripleCount(state)

29

})

30

31

32

33

Selectors can compute derived data, allowing Redux to store the minimal possible state.

34

Selectors are efficient. A selector is not recomputed unless one of its arguments change.

35

Selectors are composable. They can be used as input to other selectors.

36

https://github.com/reactjs/reselect \*/

37

1

import { connect } from 'react-redux'

2

+import { increment, doubleAsync } from '../modules/dashboard'

3

4

/\* This is a container component. Notice it does not contain any JSX,

5

nor does it import React. This component is «only» responsible for

6

wiring in the actions and state necessary to render a presentational

7

+ component - in this case, the dashboard: \*/

8

9

+import Dashboard from 'components/Dashboard'

10

11

/\* Object of action creators (can also be function that returns object).

12

Keys will be passed as props to presentational components. Here we are

13

14

@@ -10,21 +10,21 @@ const mapActionCreators = {

15

16

}

17

18

const mapStateToProps = (state) => ({

19

+ dashboard: state.dashboard

20

})

21

22

/\* Note: mapStateToProps is where you should use 'reselect' to create selectors, ie:

23

24

import { createSelector } from 'reselect'

25

+ const dashboard = (state) => state.dashboard

26

+ const tripleCount = createSelector(dashboard, (count) => count \* 3)

27

const mapStateToProps = (state) => ({

28

+ dashboard: tripleCount(state)

29

})

30

31

32

33

Selectors can compute derived data, allowing Redux to store the minimal possible state.

34

Selectors are efficient. A selector is not recomputed unless one of its arguments change.

35

Selectors are composable. They can be used as input to other selectors.

36

https://github.com/reactjs/reselect \*/

37

REACTJS

REACT NATIVE

BLOG

COMMUNITY

Changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/index.js

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com

14 src/routes/Dashboard/index.js

... @@ -1,24 +1,24 @@

1 import { injectReducer } from '../store/reducers'

2

3 export default (store) => ({

4 - path: 'counter',

5 /\* Async getComponent is only invoked when route matches \*/

6 getComponent (nextState, cb) {

7 /\* Webpack - use 'require.ensure' to create a split point

8 and embed an async module loader (jsonp) when bundling \*/

9 require.ensure([], (require) => {

10 /\* Webpack - use require callback to define

11 dependencies for bundling \*/

12 - const Counter = require('./containers/CounterContainer').default

13 - const reducer = require('./modules/counter').default

14

15 - /\* Add the reducer to the store on key 'counter' \*/

16 - injectReducer(store, { key: 'counter', reducer })

17

18 /\* Return getComponent \*/

19 - cb(null, Counter)

20

21 /\* Webpack named bundle \*/

22 - }, 'counter')

23 } }

24

1 import { injectReducer } from '../store/reducers'

2

3 export default (store) => ({

4 + path: 'dashboard',

5 /\* Async getComponent is only invoked when route matches \*/

6 getComponent (nextState, cb) {

7 /\* Webpack - use 'require.ensure' to create a split point

8 and embed an async module loader (jsonp) when bundling \*/

9 require.ensure([], (require) => {

10 /\* Webpack - use require callback to define

11 dependencies for bundling \*/

12 + const Dashboard = require('./containers/DashboardContainer').default

13 + const reducer = require('./modules/dashboard').default

14

15 + /\* Add the reducer to the store on key 'dashboard' \*/

16 + injectReducer(store, { key: 'dashboard', reducer })

17

18 /\* Return getComponent \*/

19 + cb(null, Dashboard)

20

21 /\* Webpack named bundle \*/

22 + }, 'dashboard')

23 } }

24

Renaming and changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/modules/counter.js → src/routes/Dashboard/modules/dashboard.js

12 src/routes/Dashboard/modules/counter.js → src/routes/Dashboard/modules/dashboard.js

... @@ -1,14 +1,14 @@

1 // -----

2 // Constants

3 // -----

4 -export const COUNTER\_INCREMENT = 'COUNTER\_INCREMENT'

5

6 // -----

7 // Actions

8 // -----

9 export function increment (value = 1) {

10 return {

11 - type: COUNTER\_INCREMENT,

12 payload: value

13 }

14 }

15

16 @@ -18,14 +18,14 @@ export function increment (value = 1) {

17 creating async actions, especially when combined with redux-thunk!

18

19 NOTE: This is solely for demonstration purposes. In a real application,

20 you'd probably want to dispatch an action of COUNTER\_DOUBLE and let the

21 reducer take care of this logic. \*/

22

23

24 export const doubleAsync = () => {

25 return (dispatch, getState) => {

26 return new Promise((resolve) => {

27 setTimeout(() => {

28 - dispatch(increment(getState()).counter)

29 resolve()

30 }, 200)

31 })

32 }

33

34 @@ -41,14 +41,14 @@ export const actions = {

35

36 // Action Handlers

37 // -----

38 const ACTION\_HANDLERS = {

39 [COUNTER\_INCREMENT]: (state, action) => state + action.payload

40 }

41

42 // -----

43 // Reducer

44 // -----

45 const initialState = 0

1 // -----

2 // Constants

3 // -----

4 +export const DASHBOARD\_INCREMENT = 'DASHBOARD\_INCREMENT'

5

6 // -----

7 // Actions

8 // -----

9 export function increment (value = 1) {

10 return {

11 + type: DASHBOARD\_INCREMENT,

12 payload: value

13 }

14 }

15

16 @@ -18,14 +18,14 @@ export function increment (value = 1) {

17 creating async actions, especially when combined with redux-thunk!

18

19 NOTE: This is solely for demonstration purposes. In a real application,

20 you'd probably want to dispatch an action of DASHBOARD\_DOUBLE and let the

21 reducer take care of this logic. \*/

22

23

24 export const doubleAsync = () => {

25 return (dispatch, getState) => {

26 return new Promise((resolve) => {

27 setTimeout(() => {

28 + dispatch(increment(getState()).dashboard)

29 resolve()

30 }, 200)

31 })

32 }

33

34 @@ -41,14 +41,14 @@ export const actions = {

35

36 // Action Handlers

37 // -----

38 const ACTION\_HANDLERS = {

39 + [DASHBOARD\_INCREMENT]: (state, action) => state + action.payload

40 }

41

42 // -----

43 // Reducer

44 // -----

45 const initialState = 0

REACTJS   REACT NATIVE   BLOG   COMMUNITY

Changes in (you can click the diffs image to make it larger):  
src/routes/index.js

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips SPONSORED  
Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com



src/routes/index.js

```

 2 import CoreLayout from '../layouts/CoreLayout/CoreLayout'
 3 import Home from './Home'
 4 import CounterRoute from './Counter'
 5
 6 /* Note: Instead of using JSX, we recommend using react-router
 7    PlainRoute objects to build route definitions. */
 8
 9 @@ -11,7 +12,8 @@ export const createRoutes = (store) => ({
10   component: CoreLayout,
11   indexRoute: Home,
12   childRoutes: [
13     - CounterRoute(store)
14   ]
15 })
16
17
18
19
 2 import CoreLayout from '../layouts/CoreLayout/CoreLayout'
 3 import Home from './Home'
 4 import CounterRoute from './Counter'
 5 +import DashboardRoute from './Dashboard'
 6
 7 /* Note: Instead of using JSX, we recommend using react-router
 8    PlainRoute objects to build route definitions. */
 9
10
11 component: CoreLayout,
12 indexRoute: Home,
13 childRoutes: [
14   + CounterRoute(store),
15   + DashboardRoute(store)
16 ]
17 })
18
19

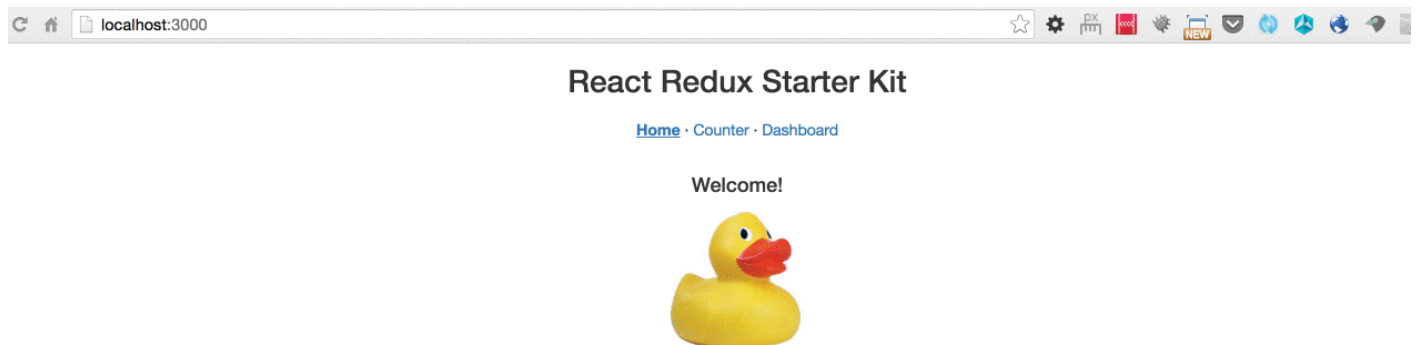
```

Here above you need to modify the routes/index.js so we will add the dashboard route.

After you run the project with:

```
npm run start
```

You shall be able to find the below app running:



As you can find above, there are two different routes with different reducers but with exactly the same feature - the counter has different number on both routes.

REACTJS | REACT NATIVE | BLOG | COMMUNITY

All the above screenshots were made on that commit:

<https://github.com/przeor/ReactC/commit/29aad0775fd8b14eeeba519e7080f5871e881f4e>

## Next steps in implementing our dashboard

In the **Dashboard.js** file we will do some little improvements in order to progress the de

Changes in (you can click the diffs image to make it larger):  
src/components/Dashboard/Dashboard.js

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com

13 src/components/Dashboard/Dashboard.js Save

@@ -4,26 +4,17 @@ import classes from './Dashboard.scss'

4 export const Dashboard = (props) => {

5 <div>

6 <h2 className={classes.dashboardContainer}>

7 - Dashboard:

8 { ' ' }

9 <span className={classes['dashboard--green']}>

10 {props.dashboard}

11 </span>

12 </h2>

13 - <button className='btn btn-default' onClick={props.increment}>

14 - Increment

15 - </button>

16 - { ' ' }

17 - <button className='btn btn-default' onClick={props.doubleAsync}>

18 - Double (Async)

19 - </button>

20 </div>

21 }

22

23 Dashboard.propTypes = {

24 - dashboard: React.PropTypes.number.isRequired,

25 - doubleAsync: React.PropTypes.func.isRequired,

26 - increment: React.PropTypes.func.isRequired

27 }

28

29 export default Dashboard

4 export const Dashboard = (props) => {

5 <div>

6 <h2 className={classes.dashboardContainer}>

7 + Dashboard visits:

8 { ' ' }

9 <span className={classes['dashboard--green']}>

10 {props.dashboard}

11 </span>

12 </h2>

13 </div>

14 }

15

16 Dashboard.propTypes = {

17 + dashboard: React.PropTypes.number.isRequired

18 }

19

20 export default Dashboard

Below we have introduced statefull DashboardContainer - we need it to do this way as we are using componentDidMount for invoking the function called **this.props.dashboardVisitIncrement()**.

**IMPORTANT:** The statefull DashboardContainer component is required here because our feature (increment by one on every visit of the dashboard route) require using a componentDidMount and we assume that in the components directory we want to have only "dumb components". In the src/components we will keep only stateless components (in other words they are also called dumb components).

Changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/containers/DashboardContainer.js

39 src/routes/Dashboard/containers/DashboardContainer.js View

@@ -1,38 +1,27 @@

1 import { connect } from 'react-redux'

2 -import { increment, doubleAsync } from '../modules/dashboard'

3 -

4 -/\* This is a container component. Notice it does not contain any JSX,

5 - nor does it import React. This component is wholly responsible for

6 - wiring in the actions and state necessary to render a presentational

7 - component - in this case, the dashboard: \*/

8 -

9 import Dashboard from 'components/Dashboard'

10

11 -/\* Object of action creators (can also be function that returns object).

12 - You will be passed an array of stateful components. You will

1 import React from 'react'

2 import { connect } from 'react-redux'

3 +import { dashboardVisitIncrement } from '../modules/dashboard'

4

5 import Dashboard from 'components/Dashboard'

REACTJS

REACT NATIVE

BLOG

COMMUNITY

16 - increment: () => increment(),

17 - doubleAsync

18 }

19

20 const mapStateToProps = (state) => ({

21 dashboard: state.dashboard

22 })

23

24 -/\* Note: mapStateToProps is where you should use 'reselect' to create selectors, i.e:

25

26 - import { createSelector } from 'reselect'

27 - const dashboard = (state) => state.dashboard

28 - const tripleCount = createSelector(dashboard, (count) => count \* 3)

29 - const mapStateToProps = (state) => ({

30 dashboard: tripleCount(state)

31 })

32

33 - Selectors can compute derived data, allowing Redux to store the minimal possible state.

34 - Selectors are efficient. A selector is not recomputed unless one of its arguments change.

35 - Selectors are composable. They can be used as input to other selectors.

36 - https://github.com/reactjs/reselect \*/

37

38 -export default connect(mapStateToProps, mapActionCreators)(Dashboard)

1 + dashboardVisitIncrement: () => dash

2

3 }

4

5

6

7

8

9

10 const mapStateToProps = (state) => ({

11 dashboard: state.dashboard

12 })

13

14

15 +class DashboardContainer extends React

16 + componentDidMount() {

17 + this.props.dashboardVisitIncrement

18 + }

19

20 + render() {

21 + return (

22 + <Dashboard {...this.props} />

23 + )

24 + }

25 + }

26

27 +export default connect(mapStateToProps, mapStateToProp

In the modules/dashboard.js we have made some improvements and cleanup of unnece  
the Counter:

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

http://reactjs.co/

15/34



Changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/modules/dashboard.js

```
29 src/routes/Dashboard/modules/dashboard.js
+++ ~1,47 ~1,28 @@
1 //
2 // Constants
3 //
4 -export const DASHBOARD_INCREMENT = 'DASHBOARD_INCREMENT'
5
6 //
7 // Actions
8 //
9 -export function increment (value = 1) {
10   return {
11     type: DASHBOARD_INCREMENT,
12     payload: value
13   }
14 }
15
16 /* This is a thunk, meaning it is a function that immediately
17  * returns a function for lazy evaluation. It is incredibly useful for
18  * creating async actions, especially when combined with redux-thunk!
19  *
20  * NOTE: This is solely for demonstration purposes. In a real application,
21  * you'd probably want to dispatch an action of DASHBOARD_DOUBLE and let the
22  * reducer take care of this logic. */
23
24 -export const doubleAsync = () => {
25   return (dispatch, getState) => {
26     return new Promise((resolve) => {
27       setTimeout(() => {
28         dispatch(increment(getState().dashboard))
29         resolve()
30       }, 200)
31     })
32   }
33 }
34
35 export const actions = {
36   increment,
37   doubleAsync
38 }
39
40 //
41 // Action Handlers
42 //
43 const ACTION_HANDLERS = {
44   [DASHBOARD_INCREMENT]: (state, action) => state + action.payload
45 }
46
47 //
48 //
49 //
50
51 //
52 // Constants
53 //
54 +export const DASHBOARD_VISITS_COUNT = 'DASHBOARD_VISITS_COUNT'
55
56 //
57 // Actions
58 //
59 +export function dashboardVisitIncrement (value = 1) {
60   return {
61     type: DASHBOARD_VISITS_COUNT,
62     payload: value
63   }
64 }
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2
```



Changes in (you can click the diffs image to make it larger):  
src/components/Dashboard/Dashboard.js

15 src/components/Dashboard/Dashboard.js View

... @@ -1,20 +1,27 @@	
1 import React from 'react'	1 import React from 'react'
2 import classes from './Dashboard.scss'	2 import classes from './Dashboard.scss'
3	3
4 -export const Dashboard = (props) => {	4 +export const Dashboard = (props) => {
	5 +
	6 + const listJSX = props.dashboard.dashboardItems.map((item, i) => {
	7 + return <h4 key={i}>{item.label}</h4>
	8 + }}
	9 +
	10 + return (
5 <div>	11 <div>
6 <h2 className={classes.dashboardContainer}>	12 <h2 className={classes.dashboardContainer}>
7 Dashboard visits:	13 Dashboard visits:
8 { ' ' }	14 { ' ' }
9 <span className={classes['dashboard--green']}>	15 <span className={classes['dashboard--green']}>
10 (props.dashboard)	16 {props.dashboard.visitsCount}
11 </span>	17 </span>
12 </h2>	18 </h2>
	19 + {listJSX}
13 </div>	20 </div>
14 -)	21 +)}
15	22
16 Dashboard.propTypes = {	23 Dashboard.propTypes = {
17 - dashboard: React.PropTypes.number.isRequired	24 + dashboard: React.PropTypes.object.isRequired
18 }	25 }
19	26
20 export default Dashboard	27 export default Dashboard

www.reactjs.co

Above you need to add a listJSX mapping code with use of:

```
const listJSX = props.dashboard.dashboardItems.map((item, i) => {  
  return <h4 key={i}>{item.label}</h4>  
})
```

and then modify the:

```
{props.dashboard.visitsCount}
```

... all that changes above are required because we have modified our reducer's structure (as you can find below):

Changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/modules/dashboard.js

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io


Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

21  src/routes/Dashboard/modules/dashboard.js Save

<pre> 13  } 14  } 15 16  - 17  export const actions = { 18    dashboardVisitIncrement 19  } 20 21  @@ -22,15 +21,27 @@ export const actions = { 22    // Action Handlers 23    // ----- 24    const ACTION_HANDLERS = { 25      - [DASHBOARD_VISITS_COUNT]: (state, action) =&gt; state + action.payload 26 27    } 28 29    // ----- 30    // Reducer 31    // ----- 32    -const initialState = 0 33 34 35  export default function dashboardReducer (state = initialState, action) { 36    const handler = ACTION_HANDLERS[action.type] 37    - 38    return handler ? handler(state, action) : state 39    -} </pre>	<pre> 13  } 14  } 15 16  export const actions = { 17    dashboardVisitIncrement 18  } 19 20 21  // Action Handlers 22  // ----- 23  const ACTION_HANDLERS = { 24    + [DASHBOARD_VISITS_COUNT]: (state, action) =&gt; { 25      + return Object.assign({}, state, { 26        + visitsCount: state.visitsCount + action.payload 27      + }) 28    + } 29  } 30 31  // ----- 32  // Reducer 33  // ----- 34  +const initialState = { 35    + visitsCount: 0, 36    + dashboardItems: [ 37      + {key: 0, label: 'Angular'}, 38      + {key: 1, label: 'jQuery'}, 39      + {key: 2, label: 'Polymer'}, 40      + {key: 3, label: 'ReactJS'} 41    + ] 42    + } 43  + 44  export default function dashboardReducer (state = initialState, action) { 45    const handler = ACTION_HANDLERS[action.type] 46 47    return handler ? handler(state, action) : state 48    +} </pre>
--	--

As you can find above we have changed the old initialState:

```
// old dashboard reducer structure
const initialState = 0
```

with the new code:

```
// new dashboard reducer structure
const initialState = {
  visitsCount: 0,
  dashboardItems: [
    {key: 0, label: 'Angular'},
    {key: 1, label: 'jQuery'},
    {key: 2, label: 'Polymer'},
    {key: 3, label: 'ReactJS'}
  ]
}
```


REACTJS | REACT NATIVE | BLOG | COMMUNITY

So generally, we have changed simple integer initialState to an object, as you can find at DASHBOARD\_VISITS\_COUNT action:

```
// old codebase
[DASHBOARD_VISITS_COUNT]: (state, action) => state + action.payload
```

with new:

```
// updated codebase
[DASHBOARD_VISITS_COUNT]: (state, action) => {
  return Object.assign({}, state, {
    visitsCount: state.visitsCount + action.payload
  })
}
```

 **Tags Saved**
...

reactjs x

js x

javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

**Our First 50,000 Stars**

facebook.github.io

 Save

**From Backbone Views To React**

leoasis.github.io

 Save

 Philips

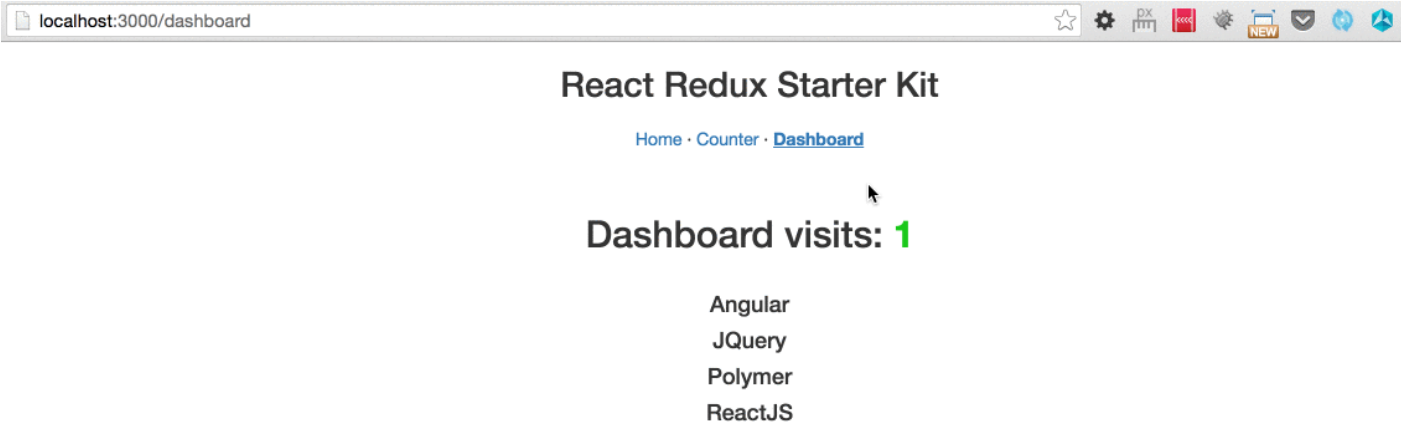
SPONSORED

**Why You Should Worry About Those Sleepless Nights**

paidpost.nytimes.com

That **DASHBOARD\_VISITS\_COUNT** change was required because as was written earlier, structure of the dashboard reducer (so we need also to change the function that handles it). The `Object.assign`'s `visitsCount` is used the same way as on the Redux's tutorial's videos on EggHead.

As a final currently, we have a little improved dashboard with a list and the `visitsCount` also works as previously (alongside with improved dashboard's reducer):



Source code from the screenshots: <https://github.com/przeor/ReactC/commit/1b2df4cd38872810257008d82559e8a0116f011b>

## Add/edit item on the dashboard list

First thing to do is to prepare the action and handlers in the modules for **DASHBOARD\_ADD\_ITEM** and **DASHBOARD\_EDIT\_ITEM**:

Changes in (you can click the diffs image to make it larger):  
`src/routes/Dashboard/modules/dashboard.js`

40 src/routes/Dashboard/modules/dashboard.js

2 // Constants

3 export const DASHBOARD\_VISITS\_COUNT = 'DASHBOARD\_VISITS\_COUNT'

4

5

6 // Actions

7

8 export function dashboardVisitIncrement (value = 1) {

9

10 }

11

12 }

13

14 }

15

16 export const actions = {

2 // Constants

3 export const DASHBOARD\_VISITS\_COUNT = 'DASHBOARD\_VISITS\_COUNT'

4 export const DASHBOARD\_ADD\_ITEM = 'DASHBOARD\_ADD\_ITEM'

5 export const DASHBOARD\_EDIT\_ITEM = 'DASHBOARD\_EDIT\_ITEM'

6

7

8 // Actions

9

10

11

12 }

13

14 }

15

16 export function dashboardAddItem (value) {

18 }

19

20

21 // Action Handlers

22

23

24

25 return Object.assign({}, state, {

26 visitsCount: state.visitsCount + action.payload

27 })

28

29 }

30

31

32

33

34

35

36

37

38 return Object.assign({}, state, {

39 visitsCount: state.visitsCount +

40 })

41

42

43 [DASHBOARD\_ADD\_ITEM]: (state, action) => {

44 const newId = Math.floor(Date.now() \* Math.random())

45 const newItem = {

46 label: action.payload,

47 id: newId,

48 };

49 return Object.assign({}, state, {

50 dashboardItems: [...state.dashboardItems, newItem]

51 })

52 [DASHBOARD\_EDIT\_ITEM]: (state, action) => {

53 const newLabel = action.payload

54 const index = action.payload.index

55 let immutableDashboardItems = [...state.immutableDashboardItems]

56 immutableDashboardItems[index].label = newLabel

57 return Object.assign({}, state, {

58 dashboardItems: immutableDashboardItems

59 })

60 }

61 }

62 }

63

64

65

66

67

68

69

70

71

72 (key: 3, label: 'ReactJS')

73 }

74 }

75

76 export default function dashboardReducer (state = initialState, action) {

77 const handler = ACTION\_HANDLERS[action.type]

78 return handler ? handler(state, action) : state

79 }

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

As you can find above, we have added and exported two new actions:



```
export function dashboardAddItem (value) {
  return {
    type: DASHBOARD_ADD_ITEM,
    payload: value
  }
}

export function dashboardEditItem (value) {
  return {
    type: DASHBOARD_EDIT_ITEM,
    payload: value
  }
}
```

... then after an action is requested from a Dashboard's component, we handle it with:

```
[DASHBOARD_ADD_ITEM]: (state, action) => {
  const mockedId = Math.floor(Date.now() / 1000)
  const newItem = {
    label: action.payload,
    id: mockedId
  }
  return Object.assign({}, state, {
    dashboardItems: [...state.dashboardItems, newItem]
  })
},
[DASHBOARD_EDIT_ITEM]: (state, action) => {
  const newLabel = action.payload.val
  const index = action.payload.editedItemIndex
  let immutableDashboardItems = [...state.dashboardItems]
  immutableDashboardItems[index].label = newLabel
  return Object.assign({}, state, {
    dashboardItems: immutableDashboardItems
  })
}
```

In both functions above, we are returning a new object with **return Object.assign**. Rest of the code shall be self-explaining for you.

Changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/containers/DashboardContainer.js

REACTJS | REACT NATIVE | BLOG | COMMUNITY

```
1 import React from 'react'
2 import { connect } from 'react-redux'
3 -import { dashboardVisitIncrement } from '../modules/dashboard'
4
5 import Dashboard from 'components/Dashboard'
6
7 -const mapActionCreators = {
8   - dashboardVisitIncrement: () => dashboardVisitIncrement(1)
9 }
10
11 -const mapStateToProps = (state) => ({
12   - dashboardVisitIncrement: () => dashboardVisitIncrement(1)
13 })
14
15 -const mapStateToProps = (state) => ({
16   - dashboardVisitIncrement: () => dashboardVisitIncrement(1)
17 })
```

Above we have simply added the functions that were created in the Dashboard/modules **dashboardAddItem** and **dashboardEditItem** .. and continuation of the same file below

continuation of src/routes/Dashboard/containers/DashboardContainer.js

**Tags Saved**

EXPLORE POCKET'S RECOMMENDATIONS:

**Our First 50,000 Stars**  
facebook.github.io

Save

**From Backbone Views To React**  
leoasis.github.io

Save

**Philips** SPONSORED ✓

**Why You Should Worry About Those Sleepless Nights**  
paidpost.nytimes.com

Save

```

14 class DashboardContainer extends React.Component {
15
16   componentDidMount() {
17     this.props.dashboardVisitIncrement();
18   }
19
20   render() {
21     return (
22       <Dashboard {...this.props} />
23     );
24   }
25 }
26
27 class DashboardContainer extends React.
28   constructor(props) {
29     super(props)
30
31     this.inputOnChange = this.inputOnChange.bind(this)
32     this.onSubmit = this.onSubmit.bind(this)
33     this.itemOnEdit = this.itemOnEdit.bind(this)
34
35     this.state = {
36       inputValue: '',
37       editedItemIndex: null
38     }
39
40   }
41
42   componentDidMount() {
43     this.props.dashboardVisitIncrement();
44   }
45
46   inputOnChange(e) {
47     this.setState({ inputValue: e.target.value })
48   }
49
50   itemOnEdit(itemIndex) {
51     const editedItem = this.props.dashboard.dashboardItems[itemIndex]
52     this.setState({ inputValue: editedItem.label, editedItemIndex: itemIndex })
53   }
54
55   onSubmit(e) {
56     e.preventDefault()
57     const val = this.state.inputValue
58     const editedItemIndex = this.state.editedItemIndex
59     if(val && editedItemIndex !== null) {
60       this.props.dashboardEditItem({ val, editedItemIndex })
61       this.setState({ inputValue: '', editedItemIndex: null })
62     } else if(val) {
63       this.props.dashboardAddItem(val)
64       this.setState({ inputValue: '' })
65     } else {
66       alert('Value can't be empty')
67     }
68   }
69
70   render() {
71     return (
72       <Dashboard {...this.props}
73         editedItemIndex={this.state.editedItemIndex}
74         itemOnEdit={this.itemOnEdit}
75         inputValue={this.state.inputValue}
76         inputOnChange={this.inputOnChange}
77         onSubmit={this.onSubmit} />
78     );
79   }
80 }

```

On the above code base, we have added new functions called `inputOnChange`, `onSubmit` and `itemOnEdit` so we need improve our constructor as well:

```

constructor(props) {
  super(props)

  this.inputOnChange = this.inputOnChange.bind(this)
  this.onSubmit = this.onSubmit.bind(this)
  this.itemOnEdit = this.itemOnEdit.bind(this)

  this.state = {
    inputValue: '',
    editedItemIndex: null
  }
}

```

... the `inputValue` keeps the current value of a text input. In the `editedItemIndex` we keep an ID of currently edited item, if there is none in edit then we make it **null**.

Later we handle all the functions logic with:

```

}

itemOnEdit(itemIndex) {
  const editedItem = this.props.dashboard.dashboardItems[itemIndex]
  this.setState({ inputValue: editedItem.label, editedItemIndex: itemIndex })
}

onSubmit(e) {
  e.preventDefault()
  const val = this.state.inputValue
  const editedItemIndex = this.state.editedItemIndex
  if(val && editedItemIndex !== null) {
    this.props.dashboardEditItem({ val, editedItemIndex })
    this.setState({ inputValue: '', editedItemIndex: null })
  } else if(val) {
    this.props.dashboardAddItem(val)
    this.setState({ inputValue: '' })
  } else {
    alert('Value can't be empty')
  }
}

```

REACTJS | REACT NATIVE | BLOG | COMMUNITY

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

```

    }
  }
}

```

There isn't nothing fancy so I won't describe too much in details (if you think it requires more detailed description then please mail us at [kamil.przeorski@gmail.com](mailto:kamil.przeorski@gmail.com)).

Final part is the **render method**:

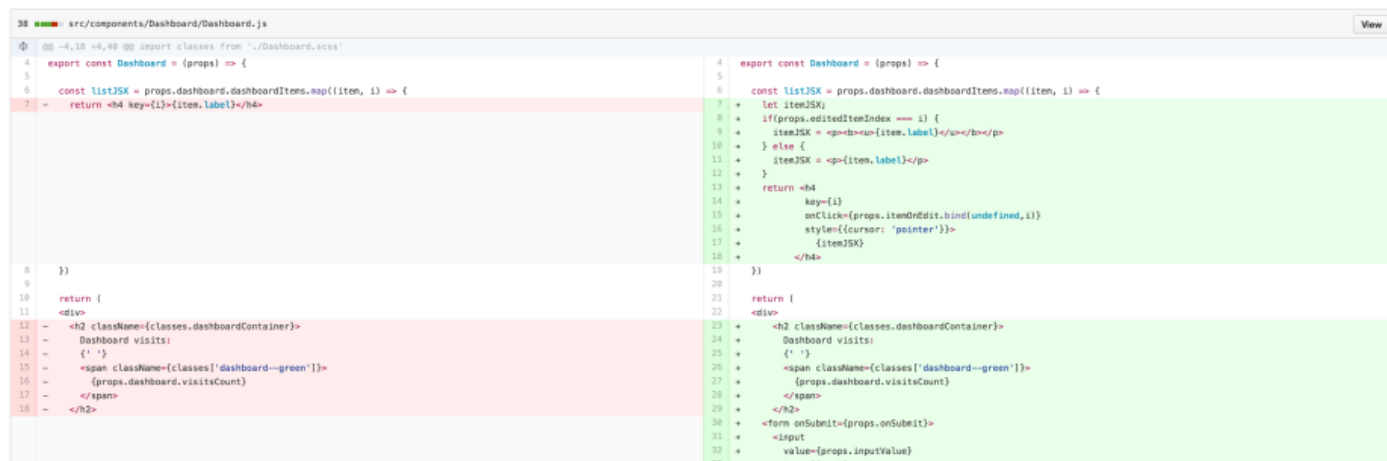
```

render () {
  return (
    <Dashboard {...this.props}
      editedItemIndex={this.state.editedItemIndex}
      itemOnEdit={this.itemOnEdit}
      inputValue={this.state.inputValue}
      inputOnChange={this.inputOnChange}
      onSubmit={this.onSubmit} />
  );
}

```

... we are passing some callbacks as **this.inputOnChange**, **this.onSubmit** and **this.itemOnEdit** - those callbacks are required to send to our dashboard "dumb component".

Changes in (you can click the diffs image to make it larger):  
[src/components/Dashboard/Dashboard.js](#)



REACTJS | REACT NATIVE | BLOG | COMMUNITY



Above, we have improved our listJSX map function:

```

const listJSX = props.dashboard.dashboardItems.map((item, i) => {
  let itemJSX;
  if(props.editedItemIndex === i) {
    itemJSX = <p><b><u>{item.label}</u></b></p>
  } else {
    itemJSX = <p>{item.label}</p>
  }
  return <h4
    key={i}
    onClick={props.itemOnEdit.bind(undefined,i)}
    style={{cursor: 'pointer'}}>
    {itemJSX}
  </h4>
}

```

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About  
Those Sleepless Nights

paidpost.nytimes.com

```

    </h4>
  })

```

 Save

... so now it make an edited item bold and underlined.

Next we have improved the render function:

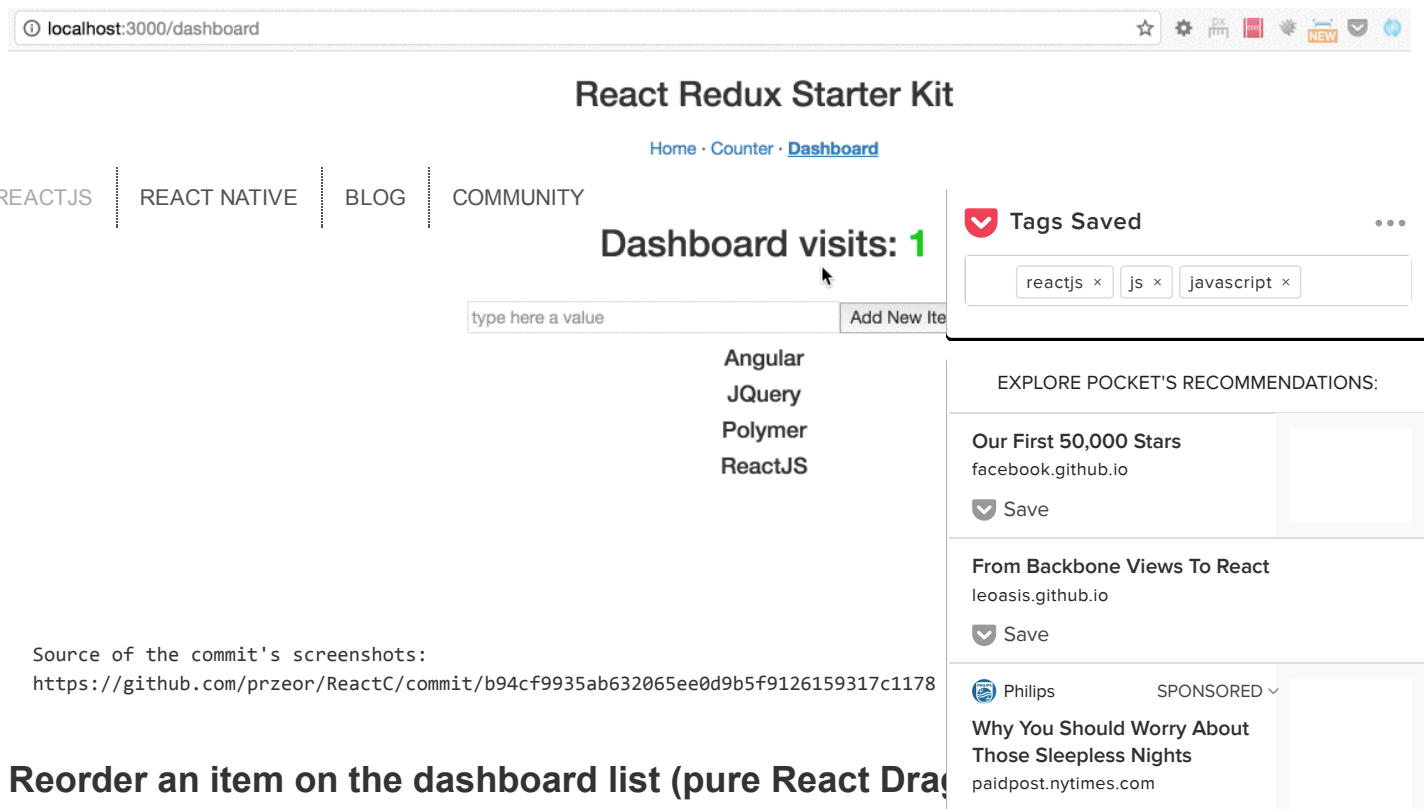
```

return (
  <div>
    <h2 className={classes.dashboardContainer}>
      Dashboard visits:
      { ' ' }
      <span className={classes['dashboard--green']}>
        {props.dashboard.visitsCount}
      </span>
    </h2>
    <form onSubmit={props.onSubmit}>
      <input
        value={props.inputValue}
        type='input'
        placeholder='type here a value'
        style={{width: 300}}
        onChange={props.inputOnChange} />
      <input
        type='submit'
        value={ props.editedItemIndex === null ? 'Add New Item To The List' : 'Edit Item' } />
    </form>
    {listJSX}
  </div>
)

```

We have added standard form and inputs - they are communicating with DashboardContainer with use of callbacks (the **value={props.inputValue}** and **onChange={props.inputOnChange}** take care of it). The submit button value is determined by the `props.editedItemIndex` - so if it is a null then that means that a user hasn't clicked any item, yet.

This is the end results of all the changes we've made above:



Source of the commit's screenshots:  
<https://github.com/przeor/ReactC/commit/b94cf9935ab632065ee0d9b5f9126159317c1178>

## Reorder an item on the dashboard list (pure React Drag

We will implement the reordering in a proper React way without using any external reordering components or libraries.

Let's start from the reducers and actions that are related to the reordering feature:

Changes in the server if the user has provided correct or incorrect (you can click the diff's image to make it larger):  
src/routes/Dashboard/modules/dashboard.js

```
37 src/routes/Dashboard/modules/dashboard.js
4 export const DASHBOARD_VISITS_COUNT = 'DASHBOARD_VISITS_COUNT'
5 export const DASHBOARD_ADD_ITEM = 'DASHBOARD_ADD_ITEM'
6 export const DASHBOARD_EDIT_ITEM = 'DASHBOARD_EDIT_ITEM'
7
8 // Actions
9
10 export function dashboardEditItem (value) {
11   // ...
12 }
13
14 // Action Handlers
15 const ACTION_HANDLERS = {
16   [DASHBOARD_ADD_ITEM]: (state, action) => {
17     // ...
18   },
19   [DASHBOARD_EDIT_ITEM]: (state, action) => {
20     // ...
21   },
22   [DASHBOARD_REORDER_ITEM]: (state, action) => {
23     // ...
24   }
25 }
26
27 return Object.assign({}, state, {
28   dashboardItems: immutableDashboardItems
29 })
30
31 // ...
32
33 // ...
34
35 // ...
36
37 // ...
38
39 // ...
40
41 // ...
42
43 // ...
44
45 // ...
46
47 // ...
48
49 // ...
50
51 // ...
52
53 // ...
54
55 // ...
56
57 // ...
58
59 // ...
60
61 // ...
62
63 // ...
64
65 // ...
66
67 // ...
68
69 // ...
70
71 // ...
72
73 // ...
74
75 // ...
76
77 // ...
78
79 // ...
80
81 // ...
82
83 // ...
84
85 // ...
86
87 // ...
88
89 // ...
90
91 // ...
92
93 // ...
94
95 // ...
96
97 // ...
98
99 // ...
100
```

An explanation for the code from the diff's - you need to understand what we will send as a payload (**const reorder = action.payload**) in the function below:

```
[DASHBOARD_REORDER_ITEM]: (state, action) => {
  const reorder = action.payload
  const reorderItem = state.dashboardItems[reorder.start]
  let newDashboardItems = []
  state.dashboardItems.map((item, i) => {
    if(i === reorder.start) {
      return
    }
    // an item from higher to lower place on the list or vice versa
    if(reorder.end < reorder.start) {
      if(i === reorder.end) {
        newDashboardItems.push(reorderItem)
      }
      newDashboardItems.push(item)
    } else {
      newDashboardItems.push(item)
      if(i === reorder.end) {
        newDashboardItems.push(reorderItem)
      }
    }
  })

  return Object.assign({}, state, {
    dashboardItems: newDashboardItems
  })
}
```

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com



Save

We will send an object with the following **start** and **end** properties as on the example below

```
// just an example schema of
// const reorder = action.payload
// so you can see what values are sent to the DASHBOARD_REORDER_ITEM
{
  start: parseInt(this.state.draggedItemIndex),
  end: parseInt(droppedItemId)
}
```

The **start** is a number of order in the **dashboardItems** array. The **end** property is an order number of a dropped-on-the-item div (the id of an item that a user dropped on the dragged item). We map over all the items in our array and based on the dragging data (start and end) we create a new array called **newDashboardItems**. Rest of the code shall be self-explained.

Changes in (you can click the diffs image to make it larger):  
src/routes/Dashboard/containers/DashboardContainer.js

```

44 src/routes/Dashboard/containers/DashboardContainer.js
1 import { connect } from 'react-redux'
2 import {
3   dashboardVisitIncrement,
4   dashboardAddItem,
5   dashboardEditItem
6 } from '../modules/dashboard'
7 import Dashboard from 'components/Dashboard'
8
9 const mapActionCreators = {
10   dashboardVisitIncrement: () => dashboardVisitIncrement(1),
11   dashboardAddItem: (value) => dashboardAddItem(value),
12   dashboardEditItem: (value) => dashboardEditItem(value)
13 }
14
15 const mapStateToProps = (state) => ({
16   // -25,38 -> 27,51 @@@@ DashboardContainer extends React.Component {
17     this.inputOnChange = this.inputOnChange.bind(this)
18     this.onSubmit = this.onSubmit.bind(this)
19     this.itemOnEdit = this.itemOnEdit.bind(this)
20
21     // -28
22     // -29
23     this.state = {
24       inputValue: '',
25       editedItemIndex: null
26     }
27
28     // -33
29     // -34
30     componentDidMount() {
31       this.props.dashboardVisitIncrement();
32     }
33   })
34
35   import {
36     dashboardVisitIncrement,
37     dashboardAddItem,
38     dashboardEditItem,
39     dashboardReorderItems
40 } from '../modules/dashboard'
41 import Dashboard from 'components/Dashboard'
42
43 const mapActionCreators = {
44   dashboardVisitIncrement: () => dashboardVisitIncrement(1),
45   dashboardAddItem: (value) => dashboardAddItem(value),
46   dashboardEditItem: (value) => dashboardEditItem(value),
47   dashboardReorderItems: (value) => dashboardReorderItems(value)
48 }
49
50 const mapStateToProps = (state) => ({
51   // -27 this.inputOnChange = this.inputOnChange.bind(this)
52   // -28 this.onSubmit = this.onSubmit.bind(this)
53   // -29 this.itemOnEdit = this.itemOnEdit.bind(this)
54   // -30 + this.handleDragStart = this.handleDragStart.bind(this)
55   // -31 + this.handleDrag = this.handleDrag.bind(this)
56   // -32 + this.handleDragOver = this.handleDragOver.bind(this)
57
58   // -33
59   // -34
60   this.state = {
61     inputValue: '',
62     editedItemIndex: null,
63     draggedItemIndex: null
64   }
65
66   // -40
67   // -41
68   componentDidMount() {
69     this.props.dashboardVisitIncrement();
70   }
71 })

```

Above we have added **draggedItemIndex** to the state which will be set in the **handleOnDragStart** function. Beside that we are also binding this to the **handleOnDrop** (here we handle login when a user drops a dragged div) and **handleOnDragOver** (this function is more like placeholder, when you can add more custom logic if you want make this dragging functionality more fancy).

... and continuation of the **containers/DashboardContainer.js**:

```
45 + handleDragStart (e) {
46 +   const id = e.target.id
47 +   this.setState({ draggedItemIndex: id })
48 + }
49 + }
50 +
51 + handleDragOver (e) {
52 +   e.preventDefault()
53 +   e.dataTransfer.dropEffect = 'move'; // See the section on the DataTransfer object.
```

[REACTJS](#)
[REACT NATIVE](#)
[BLOG](#)
[COMMUNITY](#)

 Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

## Our First 50,000 Stars

facebook.github.io

Save

## From Backbone Views To React

leoasis.github.io

Save

 Philips

SPONSORED ▾

## Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

From the above diffs we can find new functions related to the DnD:

```
handleOnDragStart (e) {
  const id = e.target.id
  this.setState({ draggedItemIndex: id })
}

handleOnDragOver (e) {
```

```

e.preventDefault()
e.dataTransfer.dropEffect = 'move'; // See the section on the DataTransfer object.
// You can add here more logic if required
}

handleOnDrop (e) {
  const droppedItemId = e.currentTarget.id
  let reorderVal = {
    start: parseInt(this.state.draggedItemId),
    end: parseInt(droppedItemId)
  }

  // the div ids have to be numbers to reorder correctly
  // and the start and end value has to be different (otherwise reorder is not required)
  const reorderIsCorrect = !isNaN(reorderVal.start) && !isNaN(reorderVal.end) && reorderVal.start !== reorderVal.end

  if(reorderIsCorrect) {
    this.props.dashboardReorderItems(reorderVal)
  }

  this.setState({ draggedItemId: null })
}

```

Save

The most important part to understand this code above is that the div has an id as a number (check the code from the **src/components/Dashboard/Dashboard.js** which is listed below) and that number is an order in the **newDashboardItems** array that is kept in the dashboard reducer.

Based on that assumption, we can use:

```

handleOnDragStart (e) {
  const id = e.target.id
  this.setState({ draggedItemId: id })
}

```

to set the start item. Below you can find how we get the final end **newDashboardItems** number

```

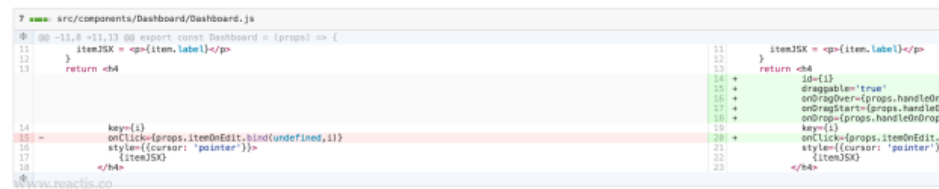
handleOnDrop (e) {
  const droppedItemId = e.currentTarget.id
  let reorderVal = {
    start: parseInt(this.state.draggedItemId),
    end: parseInt(droppedItemId)
  }
}

```

REACTJS | REACT NATIVE | BLOG | COMMUNITY

Rest of the code shall be self-explained (if something is unclear, please contact us at [kan@leoasis.io](mailto:kan@leoasis.io))

Changes in (you can click the diffs image to make it larger):  
[src/components/Dashboard/Dashboard.js](#)



```

const listJSX = props.dashboard.dashboardItems.map((item, i) => {
  let itemJSX;
  if(props.editedItemId === i) {
    itemJSX = <p><b><u>{item.label}</u></b></p>

```

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
[facebook.github.io](https://facebook.github.io)

Save

From Backbone Views To React  
[leoasis.github.io](https://leoasis.github.io)

Save

Philips

SPONSORED

Why You Should Worry About  
 Those Sleepless Nights  
[paidpost.nytimes.com](https://paidpost.nytimes.com)

```

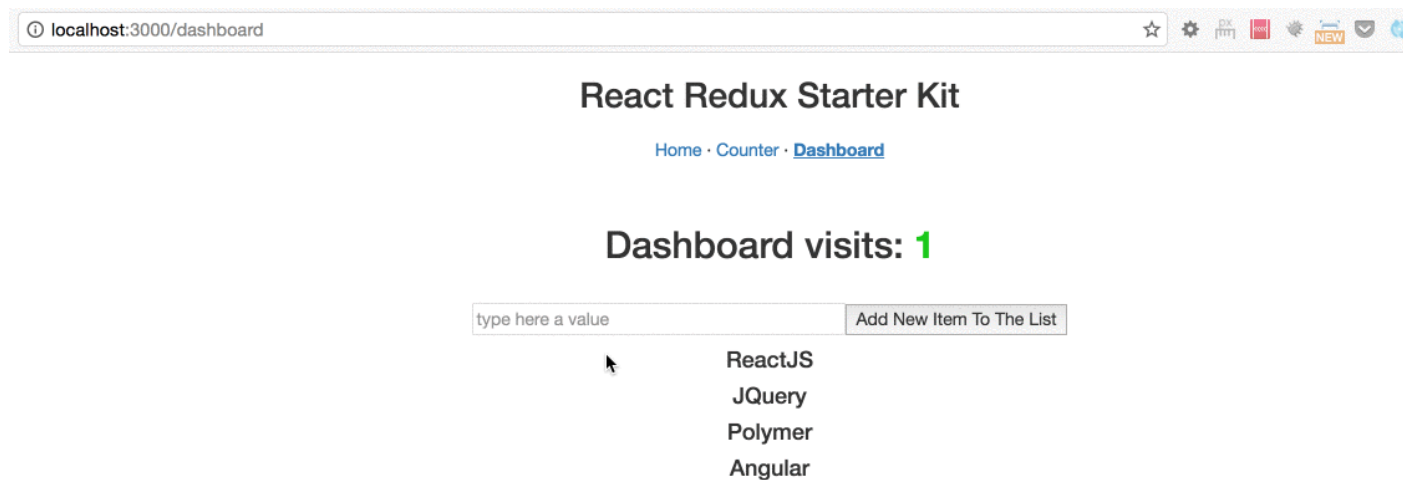
    } else {
      itemJSX = <p>{item.label}</p>
    }
    return <h4
      id={i}
      draggable='true'
      onDragOver={props.handleOnDragOver}
      onDragStart={props.handleOnDragStart}
      onDrop={props.handleOnDrop}
      key={i}
      onClick={props.itemOnEdit.bind(undefined, i)}
      style={{cursor: 'pointer'}}>
        {itemJSX}
      </h4>
    )
  })

```

Save

The last part is to add callbacks (onDragOver, onDragStart and onDrop), modify style and add the **id={i}** (so we can take a DIV's id as an information required to make DnD works).

This is how the app shall behave after that all our changes:



Source of the commit's screenshots:

<https://github.com/nrznor/ReactC/commit/1f40f87a9h0ah4hd823e7a922f88f7a8a8ah1h9a>

REACTJS | REACT NATIVE | BLOG | COMMUNITY

## Login with mocked data (front-end)

We will implement login functionality that works purely on front-end (later we will create a simple backend endpoint). First step is to prepare new session reducer which will be app-wise (it will be used all across the app).

Create a new directory at "src/modules" location  
.. and then create a new file src/modules/session.js

**IMPORTANT:** we are creating this module directory in the main src/\*\* location because that one is the main app. In the session reducer we will keep all the actions and data related to our user session (li

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com



78 src/modules/session.js

00 =&gt;0,0 +1,78 00

```

1 +import {push} from 'react-router-redux'
2 +
3 +// -----
4 +// Constants
5 +// -----
6 +export const SESSION_LOGIN_SUCCESS = 'SESSION_LOGIN_SUCCESS'
7 +export const SESSION_LOGIN_FAIL = 'SESSION_LOGIN_FAIL'
8 +
9 +// -----
10 +// Actions
11 +// -----
12 +export function loginSuccess (value) {
13 +  return {
14 +    type: SESSION_LOGIN_SUCCESS,
15 +    payload: value
16 +  }
17 +}
18 +
19 +export function loginFail (value) {
20 +  return {
21 +    type: SESSION_LOGIN_FAIL,
22 +    payload: value
23 +  }
24 +}
25 +
26 +export const loginAsync = (loginObj) => {
27 +  return async (dispatch, getState) => {
28 +    let loginToken = await new Promise((resolve) => {
29 +      setTimeout(() => {
30 +        resolve()
31 +      }, 200)
32 +    }).then(() => {
33 +
34 +      if(loginObj.user === 'przeor' && loginObj.password === 'mwp.io') {
35 +        return 'www.mwp.io' // just a mocked token
36 +      } else {
37 +        return 'invalid' // mocked non successful login
38 +      }
39 +    })
40 +
41 +    if(loginToken === 'invalid') {
42 +      dispatch(loginSuccess(loginToken))
43 +      dispatch(push('/dashboard'))
44 +    } else {
45 +      dispatch(loginFail(loginToken))
46 +    }
47 +  }
48 +}
49 +

```

Above we have created **SESSION\_LOGIN\_SUCCESS** and **SESSION\_LOGIN\_FAIL** actions.

The interesting part is the loginAsync function which looks like below:

```

export const loginAsync = (loginObj) => {
  return async (dispatch, getState) => {
    let loginToken = await new Promise((resolve) => {
      setTimeout(() => {
        resolve()
      }, 200)
    }).then(() => {

      if(loginObj.user === 'przeor' && loginObj.password === 'mwp.io') {
        return 'www.mwp.io' // just a mocked token
      } else {
        return 'invalid' // mocked non successful login
      }
    })

    dispatch(loginSuccess(loginToken))
    dispatch(push('/dashboard'))
  } else {
    dispatch(loginFail(loginToken))
  }
}
}
}

```

REACTJS | REACT NATIVE | BLOG | COMMUNITY

The function received the **loginObj** which is composed of two keys:

```

loginObj.user
loginObj.password

```

That data will be sent to the backend server (after we will unmock now), currently we only

```

if(loginObj.user === 'przeor' && loginObj.password === 'mwp.io') {

```

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io



From Backbone Views To React

leoasis.github.io



SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com

Also as you can find, we have created an asynchronous function (**return async (dispatch, getState) => {}**) which awaits on the promise resolve after 200 milliseconds timeout (currently it's a mock, later it will be real POST to the server). Then depending on if you have provided correct login details, it returns:

```
}).then(() => {
  if(loginObj.user === 'przeor' && loginObj.password === 'mwp.io') {
    return 'www.mwp.io' // just a mocked token
  } else {
    return 'invalid' // mocked non successful login
  }
})
```

The **'invalid'** means, that you have provided incorrect login details (later that will be sent back from the server).

The last executing code of the **loginAsync** is:

```
if(loginToken !== 'invalid') {
  dispatch(loginSuccess(loginToken))
  dispatch(push('/dashboard'))
} else {
  dispatch(loginFail(loginToken))
}
```

The **dispatch** comes from the **react-thunk** - that means, that the loginAsync is returned immediately and waits for lazy evaluation (in our case on a respond from the server if the user has provided correct or incorrect details) and then dispatch an action depending on the server response.

On a valid details, we dispatch two actions:

```
dispatch(loginSuccess(loginToken))
dispatch(push('/dashboard'))
```

First **dispatch** is for the **loginSuccess** with the **loginToken** value as a variable and the second the **push** which comes from the **react-router-redux (import {push} from 'react-router-redux')** - this function simply push user to **/dashboard** route if he is on a different one with use of push function from the routing librarie that we use..

REACTJS | REACT NATIVE | BLOG | COMMUNITY

```
51 //
52 // Action Handlers
53 //
54 +const ACTION_HANDLERS = {
55 + [SESSION_LOGIN_SUCCESS]: (state, act
56 +   return Object.assign({}, state,
57 +     loginToken: action.payload,
58 +     isNotLoggedIn: false
59 +   )
60 + },
61 + [SESSION_LOGIN_FAIL]: (state, act
62 +   return Object.assign({}, state,
63 +     loginToken: action.payload
64 +   )
65 + }
66 +}
67 //
68 // Reducer
69 //
70 +const initialState = {
71 +   count: 0,
72 +   isNotLoggedIn: true,
73 +   loginToken: 'none'
74 + }
75 +}
76 +export default function sessionRedu
77 + const handler = ACTION_HANDLERS[a
78 +   return handler ? handler(state, a
79 + }
```

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips SPONSORED  
Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com

... and above you can find the remaining action handlers and the session reducer is looki



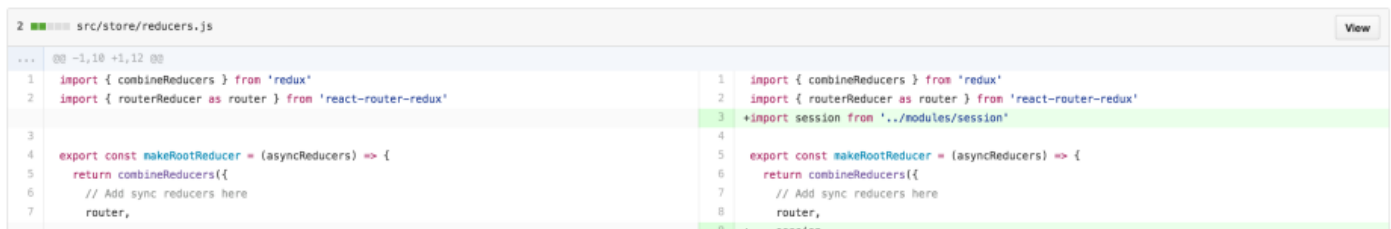
```
// -----
// Action Handlers
// -----
const ACTION_HANDLERS = {
  [SESSION_LOGIN_SUCCESS]: (state, action) => {
    return Object.assign({}, state, {
      loginToken: action.payload,
      isNotLoggedIn: false
    })
  },
  [SESSION_LOGIN_FAIL]: (state, action) => {
    return Object.assign({}, state, {
      loginToken: action.payload
    })
  }
}

// -----
// Reducer
// -----
const initialState = {
  count: 0,
  isNotLoggedIn: true,
  loginToken: 'none'
}
```

There is a flag for simplicity which handles if a user is logged in (**isNotLoggedIn**) and we keep in that reducer the token which shall be sent to the backend on each request while a user is logged in (the loginToken may come from the backend's JSON Web Token).

For now we are implementing very simple login solution and then we will build up on it to make it more powerful.

Changes in (you can click the diffs image to make it larger):  
src/store/reducers.js



REACTJS | REACT NATIVE | BLOG | COMMUNITY



Above we are simply importing the session reducer with **import session from '../modules/session'** and adding it to the combineReducers:

```
export const makeRootReducer = (asyncReducers) => {
  return combineReducers({
    // Add sync reducers here
    router,
    session,
    ...asyncReducers
  })
}
```

That's all there, then we need to improve **CoreLayout.js**:

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About  
Those Sleepless Nights

paidpost.nytimes.com



Changes in (you can click the diffs image to make it larger):  
src/layouts/CoreLayout/CoreLayout.js

```

54 src/layouts/CoreLayout/CoreLayout.js
55 -- 1,19 -1,47 00
56 -import React from 'react'
57 -import Header from '../components/Header'
58 -import classes from './CoreLayout.scss'
59 -import './styles/core.scss'
60
61 -export const CoreLayout = ({ children }) => {
62 -  <div className='container text-center'>
63 -    <header />
64 -    <div className={classes.mainContainer}>
65 -      {children}
66 -    </div>
67 -  </div>
68 -}
69 -CoreLayout.propTypes = {
70 -  children: React.PropTypes.element.isRequired
71 -}
72 -export default CoreLayout
73
74 +import React, { Component, PropTypes } from 'react'
75 +import Header from '../components/Header'
76 +import classes from './CoreLayout.scss'
77 +import './styles/core.scss'
78 +import { connect } from 'react-redux'
79 +import { loginAsync } from '../modules/session'
80
81 +const mapActionCreators = {
82 +  loginAsync
83 +}
84
85 +const mapStateToProps = (state) => ({
86 +  session: state.session
87 +})
88
89 +class CoreLayout extends Component {
90 +  static propTypes = {
91 +    children: PropTypes.element.isRequired
92 +  }
93 +  constructor(props) {
94 +    super(props)
95 +    this.handleLogin = this.handleLogin.bind(this)
96 +  }
97 +  handleLogin(loginObj, e) {
98 +    e.preventDefault()
99 +    this.props.loginAsync(loginObj)
100 +  }
101 +  render () {
102 +    return (
103 +      <div className='container text-center'>
104 +        <header />
105 +        <div className={classes.mainContainer}>
106 +          {this.props.children}
107 +        </div>
108 +      </div>
109 +    )
110 +  }
111 +}
112 +export default connect(mapStateToProps, mapActionCreators)(CoreLayout)

```

Above, we make the CoreLayout as a smart component which is connected to the session reducer with use of:

```

import React, { Component, PropTypes } from 'react'
import Header from '../components/Header'
import classes from './CoreLayout.scss'
import './styles/core.scss'
import { connect } from 'react-redux'
import { loginAsync } from '../modules/session'

const mapActionCreators = {
  loginAsync
}

const mapStateToProps = (state) => ({
  session: state.session
})

```

REACTJS | REACT NATIVE | BLOG | COMMUNITY

Then we improve the CoreLayout component as following:

```

class CoreLayout extends Component {
  static propTypes = {
    children: PropTypes.element.isRequired
  }

  constructor(props) {
    super(props)
    this.handleLogin = this.handleLogin.bind(this)
  }

  handleLogin(loginObj, e) {
    e.preventDefault()
    this.props.loginAsync(loginObj)
  }

  render () {

```

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

**Our First 50,000 Stars**  
facebook.github.io

Save

**From Backbone Views To React**  
leoasis.github.io

Save

**Why You Should Worry About Those Sleepless Nights**  
paidpost.nytimes.com

```

return (
  <div className='container text-center'>
    <Header
      handleLogin={this.handleLogin}
      session={this.props.session} />
    <div className={classes.mainContainer}>
      {this.props.children}
    </div>
  </div>
)
}
}

```

Save

```
export default connect(mapStateToProps, mapActionCreators)(CoreLayout)
```

In the CoreLayout you can find a function with handles login called **handleLogin**(the **e.preventDefault()** is a standard thing so I won't explain how it works here) which sends the user and password's object to the session reducer with use of **this.props.loginAsync(loginObj)**.

To the Header's component we send down the handleLogin and the session's reducer data (**session={this.props.session}**).

Changes in (you can click the diffs image to make it larger):  
src/components/Header/Header.js

```

@@ -2,21 +2,73 @@ import React from 'react'
1  import { IndexLink, Link } from 'react-router'
2  import classes from './Header.scss'
3
4
5  -export const Header = () => {
6  -  <div>
7  -    <h1>React Redux Starter Kit</h1>
8  -    <IndexLink to="/" activeClassName={classes.activeRoute}>
9  -      Home
10 -    </IndexLink>
11 -    {', ' . ' }
12 -    <Link to="/counter" activeClassName={classes.activeRoute}>
13 -      Counter
14 -    </Link>
15 -    {', ' . ' }
16 -    <Link to="/dashboard" activeClassName={classes.activeRoute}>
17 -      Dashboard
18 -    </Link>
19 -  </div>
20 -}
21
22  +let loginObj = {
23  +  user: '',
24  +  password: ''
25  +}
26
27  +const usernameOnChange = (e) => {
28  +  loginObj.user = e.target.value
29  +}
30
31  +const passwordOnChange = (e) => {
32  +  loginObj.password = e.target.value
33  +}
34
35  +const prepareLoginJSX = (props) => {<div>
36  +  <form onSubmit={props.handleLogin.bind(undefined, loginObj)}>
37  +    <input
38  +      type="text"
39  +      placeholder="username"
40  +      style={{width: 100}}
41  +      name="username"
42  +      onChange={usernameOnChange} />
43  +    <input
44  +      type="password"
45  +      placeholder="password"
46  +      style={{width: 100}}

```

REACTJS    REACT NATIVE    BLOG    COMMUNITY

www.reactjs.co

Above the standard thing for login forms as **prepareLoginJSX** function which returns a form. There are some on change as **onChange={usernameOnChange}** and **onChange={passwordOnChange}** which update the loginObj after a user hits the submit button, then the loginObj is sent via callback to the session reducer (**props.handleLogin.bind(undefined, loginObj)**).

Tags Saved

reactjs x   js x   javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io

Save

From Backbone Views To React  
leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com





```

37 +
38 +export const Header = (props) => {
39 +  let loginFormJSX
40 +  let loginMessageJSX = null
41 +
42 +  if(props.session.isNotLoggedIn) {
43 +    if(props.session.loginToken === 'invalid') {
44 +      loginMessageJSX = <p>Invalid login details, please try with correct user and password</p>
45 +    }
46 +
47 +    loginFormJSX = prepareLoginJSX(props)
48 +  } else {
49 +    loginFormJSX = null
50 +  }
51 +
52 +  return (
53 +    <div>
54 +      <h1>React Redux Starter Kit</h1>
55 +      <div>
56 +        <IndexLink to="/" activeClassName={classes.activeRoute}>
57 +          Home
58 +        </IndexLink>
59 +        {', ' }
60 +        <Link to="/counter" activeClassName={classes.activeRoute}>
61 +          Counter
62 +        </Link>
63 +        {', ' }
64 +        <Link to="/dashboard" activeClassName={classes.activeRoute}>
65 +          Dashboard
66 +        </Link>
67 +      </div>
68 +      {loginFormJSX}
69 +      {loginMessageJSX}
70 +    </div>
71 +  )
72 +}
73
74 export default Header

```

... and in the **export const Header = (props) => {** we simply return the login form in case if a user is not logged in **props.session.isNotLoggedIn** and in case if a user put's wrong details then we show him a message:

```

if(props.session.loginToken === 'invalid') {
  loginMessageJSX = <p>Invalid login details, please try with correct user and password</p>
}

```

Everything is done besides the Dashboard improvements:

```

9 src/routes/Dashboard/containers/DashboardContainer.js
10
11 // -16,7 +16,8 @@ const mapActionCreators = {
12 }
13
14 const mapStateToProps = (state) => ({
15   - dashboard: state.dashboard
16 })
17
18 // -58,7 +51,7 @@ class DashboardContainer extends React.Component {
19
20   handleOnDragOver (e) {
21     e.preventDefault()
22   }
23
24   - e.dataTransfer.dropEffect = 'move' // See the section on the DataTransfer object.
25   // You can add here more logic if required
26 }
27
28 // -97,6 +98,18 @@ class DashboardContainer extends React.Component {
29 }
30
31 // -16,7 +16,8 @@ const mapActionCreators = {
32 }
33
34 const mapStateToProps = (state) => ({
35   + dashboard: state.dashboard,
36   + session: state.session
37 })
38
39 // -58,7 +51,7 @@ class DashboardContainer extends React.Component {
40
41   handleOnDragOver (e) {
42     e.preventDefault()
43   }
44
45   + e.dataTransfer.dropEffect = 'move' // See the section on the DataTransfer object.
46   // You can add here more logic if required
47 }
48
49 // -97,6 +98,18 @@ class DashboardContainer extends React.Component {
50 }

```

REACTJS | REACT NATIVE | BLOG | COMMUNITY

Tags Saved

reactjs x js x javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars  
facebook.github.io



From Backbone Views To React  
leoasis.github.io



Philips SPONSORED  
Why You Should Worry About  
Those Sleepless Nights  
paidpost.nytimes.com

Above we simply add the session reducer to the DashboardContainer and then check if a user is not logged in and show him a message via render function:

```

render () {
  if(this.props.session.isNotLoggedIn) {
    return <h4>Please login in order to access your dashboard</h4>
  }
}

```

Based on that all changes we have made then you shall be able to run this app with login form and animation below:

localhost:3000/counter

Save

React Redux Starter Kit

[Home](#) · [Counter](#) · [Dashboard](#)

Counter: 0

Commits screenshots source: <https://github.com/przeor/ReactC/commit/52ac2ef6317e53ad736355506db346eab898c7e2>

## A summary (front-end part)

This is first part of the e-book related to front-end implementations.

How do you like it? What we can improve? Please mail us with your feedback.

**Do you want the backend implementation? Then we need your feedback what backend tech stack you would like to use in your future projects?**

**WHAT TO USE ON THE BACKEND IN THE React Convention™ e-book:**

- a) GraphQL + Relay
- b) FalcorJS
- c) Standalone REST API implementation with Axios on the front-end

Please mail us at [kamil.przeorski@gmail.com](mailto:kamil.przeorski@gmail.com) and based on your feedback we will continue this free book.

Here you can post any suggestions about the e-book: [ReactJS](#), [React Native](#), [GraphQL](#) developers on FB.

REACTJS | REACT NATIVE | BLOG | COMMUNITY

[github.com/przeor](#)

Follow @przeor

384

[Twitter.com/przeor](#)

Follow

Search In Google For ReactJS Convention™

Tags Saved

reactjs x

js x

javascript x

EXPLORE POCKET'S RECOMMENDATIONS:

Our First 50,000 Stars

facebook.github.io

Save

From Backbone Views To React

leoasis.github.io

Save

Philips

SPONSORED

Why You Should Worry About Those Sleepless Nights

paidpost.nytimes.com