

0x07. React Redux action creator+normalizr

Home (/)

My training planning (/my)

Front-end

JavaScript

ES6

React



Projects (/projects/current)

⚙ Weight: 1



Project QA Reviews can make corrections to review

Project QA Reviews can make corrections to review

24 6:00 AM



Evaluation quizzes (/dashboards/my_current_evaluation_quizzes)



Curriculums (/dashboards/my_curriculums)



Concepts (/concepts)



Conference rooms (/dashboards/video_rooms)



Servers (/servers)



Sandboxes (/user_containers/current)

Resources

Tools (/dashboards/my_tools)

Read or watch:



- Normalizr (/rltoken/WdUP5b9xtR2-yW2f2WsLA)
- Normalizing State Shape (/rltoken/fFyKP9fUVnyFCTOCeLUN4A)
- Redux Getting started and core concepts (/rltoken/PgQVCavA2FKKaXevbFeq4A)
- Redux Actions (/rltoken/OXROwvWYxWqt97UmKDM7lw)



- Async Actions (/rltoken/hLMWlnrGL4NAq-vZ_S560w)
- Writing tests for Redux (/rltoken/TpqsZMneviPn4OiD8PPNqg)

Peers (/users/peers)

Learning Objectives

Discord (/https://discord.com/app)

At the end of this project, you are expected to be able to explain to anyone (/rltoken/iShpotZOL-g09X6NbWUGEA), **without the help of Google**:

- Normalizr's purpose and how to use it
- schema and normalization of nested JSON
- core concepts of Redux





- Redux actions
- Redux action creators
- async actions in Redux
- how to write tests for Redux



Home(/)

Requirements



- Allowed editors: `vi`, `vim`, `emacs`, Visual Studio Code
- All your files should end with a new line
- All your files will be interpreted/compiled on Ubuntu 18.04 LTS using node 12.x.x and npm



6 Projects(/projects/current)

- A `README.md` file, at the root of the folder of the project, is mandatory
- Push all of your files, including `package.json` and `.babelrc`



- All of your functions must be exported

QA Reviews I can make(/corrections/to_review)



Provided files

Validation quizzes(/dashboards/my_current_evaluation_quizzes)

notifications.json



Curriculums(/dashboards/my_curriculums)

login-success.json



Concepts(/concepts)

Click to show/hide contents of login-success.json



Conference rooms(/dashboards/video_rooms)



Tasks

Servers(/servers)



0. Read data from a JSON

Sandboxes(/user_containers/current)

mandatory

Reuse the latest dashboard project you worked on in the React course 0x06-React_state



Tools(/dashboards/my_tools)

For this task, place `notifications.json` into the root of the project directory and use the data inside for the next step.



Video on demand(/dashboards/videos)

Create a new `notifications.js` file in a `schema` folder:

- Import the JSON data from `notifications.json` and give it a name. Try `import * as [variable name] from [path to notifications.json]`



Peers(/users/peers)

- Create a function named `getAllNotificationsByUser` that accepts `userId` as an argument
- The function should return a list containing all the `context` objects from the `notifications.json` data when the `author id` is the same as the `userId`



Discord(<https://discord.com/app>)

In the same `schema` directory, create a `notifications.test.js` file:

- Add a test that uses the id `5debd764a7c57c7839d722e9` and verifies that the following data is returned:



My Profile(/users/my_profile)

```
[
  {
    (/) guid: "2d8e40be-1c78-4de0-afc9-fcc147afd4d2",
      isRead: true,
      type: "urgent",
      value:
        Home(/)
        "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod te
mpor incididunt."
    }, My Planning(/planning/me)
    {
      guid: "280913fe-38dd-4abd-8ab6-acdb4105f922",
      isRead: false,
      type: "urgent",
      value:
        QA Reviews I can make corrections to review
        "I am maecenas sed"
    }
  ]
  ?] Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)
```

Tips:

- You can easily import JSON data using Babel
- When writing your test, you can use the `arrayContaining` method from Jest to easily compare what the function returns and what you are expecting

Requirements (/concepts)

- You can use any loop function to go through the array
- All the tests in the project should pass

Repo: Servers(/servers)

- GitHub repository: `alx-react`
- Directory: `0x07-react_redux_action_creator_normalizr`
- Sandboxes(/user_containers/current)
- File: `task_0/dashboard/src/schema/notifications.js`,
`task_0/dashboard/src/schema/notifications.test.js`

Tools(/dashboards/my_tools)

1 Normalize a nested JSON

Video on demand(/dashboards/videos)

mandatory

Copy over `dashboard` from the previous task into a `task_1` directory at the root of the project

Modify `src/schema/notifications.js` to set up a schema using Normalizr

Peers(/users/peers)

You're going to use `schema.Entity` to create a 3 of entities.

The first one is an example the task will provide for you.
Discord(<https://discord.com/app>)

```
const user = new schema.Entity("users")
```

- Create a message entity in a variable called `message` whose key is `messages` and set the `idAttribute` to the string `guid` in the options
- Create a notification entity in a variable called `notification` whose key is `notifications` and set the definition of the entity as so:

My Profile(/users/my_profile)

- author: user
- context: message



Add a test in `schema/notifications.test.js` to verify that your normalized data has a correct `result` array. It should contain:



Home(/)

"5debd76480edafc8af244228"

"5debd764507712e7a1307303"



My Planning(planning/me)

"5debd76444dd4dafea89d53b"

"5debd76485ee4dfd1284f97b"

"5debd7644e561e022d66e61a"



Projects/projects/current)

"5debd76413f0d5e5429c28a0"

"5debd7642e815cd350407777"



QA Reviews I can make(/corrections/to_review)

"5debd7646ef31e0861ec1cab"

"5debd764a4f11eabef05a81d"



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

"5debd76468cb5b277fd125f4"

"5debd764de9fa684468cdc0b"

Add a test to verify that your normalized data has a correct `users` entity. Test to access the user with the id `5debd764a7c57c7839d722e9`. It should return:



Curriculums(/dashboards/my_curriculums)

age: 25,

Concepts(/concepts)

email: "poole.sanders@holberton.nz",

id: "5debd764a7c57c7839d722e9",

name: { first: "Poole", last: "Sanders" },

Conference rooms(/dashboards/video_rooms)

picture: "http://placeholder.it/32x32"



Add a test to verify that your normalized data has a correct `messages` entity. Test to access the message with the guid `efb6c485-00f7-4fdf-97cc-5e12d14d6c41`. It should return:



Servers(servers)

guid: "efb6c485-00f7-4fdf-97cc-5e12d14d6c41",

isRead: false,

type: "default",

value: "Cursus risus at ultrices mi."



Sandboxes(/user_containers/current)



Tools(/dashboards/my_tools)

Add a test to verify that your normalized data has a correct `notifications` entity. Test to access the notification with the id `5debd7642e815cd350407777`. It should return:



Video on demand(/dashboards/videos)

author: "5debd764f8452cf92346c772",

context: "3068c575-d619-40af-bf12-dece1ee18dd3",

id: "5debd7642e815cd350407777"



Peers(/users/peers)

Tips:



Discord(<https://discord.com/app>)

- The expected goal is to obtain a very easy to use dataset
- If you are having undefined issues, look at `idAttribute` from the Normalizr documentation



Requirements:

- You must export the list of notifications using a Normalizr's `normalize`
- All my Profile (in the project profile)

Repo:

(/)

- GitHub repository: alx-react
- Directory: 0x07-react_redux_action_creator_normalizr
- File: task_1/dashboard/src/schema/notifications.js, task_1/dashboard/src/schema/notifications.test.js



My Planning(/planning/me)



2. Filter a normalized Schema

Projects(/projects/current)

mandatory

Copy the contents of dashboard from the task_1 directory into a task_2 directory at the root of the

project QA Reviews I can make(/corrections/to_review)

Modify the function getAllNotificationsByUser to use the normalized dataset



Requirements evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

- You should only use one loop at this point
- You should not use Object.keys
- You should not have to modify the test, and the test should pass correctly
- All the tests in the project should pass



Curriculums(/dashboards/my_curriculums)



Repo: Concepts(/concepts)

- GitHub repository: alx-react
- Directory: 0x07-react_redux_action_creator_normalizr
- File: task_2/dashboard/src/schema/notifications.js



Conference rooms(/dashboards/video_rooms)



Servers(/servers)



Sandboxes(/user_containers/current)

3. Create actions for the course list

mandatory



Tools(/dashboards/my_tools)

Copy the dashboard folder from the task_2 directory into a directory named task_3



Create a new folder named actions Video on demand(/dashboards/videos)

Create the action types:

In a file named courseActionTypes.js, create two action types:



- Peers(/users/peers)
- UNSELECT_COURSE



They will be used to perform actions on a specific course

Create the action creators:

In a file named courseActionCreators.js, create two action creators that will send the two types we previously created:

- The My Profile(/users/my_profile) will accept index as argument

My Profile(/users/my_profile)



- The function `unSelectCourse` will accept `index` as argument



Test the action creators:

(/)

In a file named `courseActionCreators.test.js`, write a test for the `selectCourse` action. Calling the creator with 1 as argument should return: `{ type: SELECT_COURSE, index: 1 }`



Home(/)

Write a test for the `unSelectCourse` action. Calling the creator with 1 as argument should return: `{ type: UNSELECT_COURSE, index: 1 }`



My Planning(/planning/me)



Repo:

Projects(/projects/current)

- GitHub repository: `alx-react`
- Directory: `0x07-react_redux_action_creator_normalizr`
- ✓ • File: `task_3/dashboard/src/actions/courseActionCreators.js`,
`task_3/dashboard/src/actions/courseActionCreators.test.js`,
`task_3/dashboard/src/actions/courseActionTypes.js`
- ? • Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

4. Create actions for the UI

mandatory



Curriculums(/dashboards/my_curriculums)

Copy the dashboard folder from `task_3` into a directory labeled `task_4`



`src/actions/uiActionTypes.js`, create four action types:

e.g. `export const LOGIN = "LOGIN"`



create the action types.

Conferences(/dashboards/video_rooms)

- LOGIN
- LOGOUT
- DISPLAY_NOTIFICATION_DRAWER
- HIDE_NOTIFICATION_DRAWER



Sandboxes(/user_containers/current)

They will be used to define when a user is logging in, logging out, and display / hide the notifications drawer



Tools(/dashboards/my_tools)

Create the action creator:

In a file named `uiActionCreators.js`, the goal of this section is to create four action creators that will send the four types we previously created. Remember to import all the types from `uiActionTypes` in this file.



Video on demand(/dashboards/videos)

- The function `login` will accept `email` and `password` as arguments. It will return the action with



`LOGIN` as a type and the user object:

Peers(/users/peers)

```
{ user : { email, password } }
```



Discord(<https://discord.com/app>)

- The function `logout` will create the action with the type `LOGOUT`
- The function `displayNotificationDrawer` will create the action with the type `DISPLAY_NOTIFICATION_DRAWER`
- The function `hideNotificationDrawer` will create the action with the type `HIDE_NOTIFICATION_DRAWER`

My Profile(/users/my_profile)



Test the action creators:



In a file named `uiActionCreators.test.js`, write a test for each of the action creator you wrote previously.



Home(/)

Repo:



- GitHub repository: `alx-react`
- Directory: `0x07-react_redux_action_creator_normalizr`
- File: `task_4/dashboard/src/actions/uiActionTypes.js`,
`task_4/dashboard/src/actions/uiActionCreators.js`,
`task_4/dashboard/src/actions/uiActionCreators.test.js`



Projects(/projects/current)



QA Reviews I can make(/corrections/to_review)

5. Create actions for the notification list

mandatory



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)

Copy dashboard from the `task_4` directory into `task_5`

Create the action types



`src/actions/notificationActionTypes.js` create two action types:

- `MARK_AS_READ`
- `SET_TYPE_FILTER`



Concepts(/concepts)

Create the filter states



`src/actions/notificationActionTypes.js` create a constant named `NotificationTypeFilters`, that will contain the two filter states:



- `DEFAULT`
- `URGENT`

Servers(/servers)



They will be used when the user interacts with the notification drawer

Sandboxes(/user_containers/current)

Create the action creator



Import the action types you just created in `src/actions/notificationActionTypes.js`

Tools(/dashboards/my_tools)

In a file named `notificationActionCreators.js`, create two action creators that will send the two action types we previously created:



Video on demand(/dashboards/videos)

- The function `markAsAread` will accept `index` as argument
- The function `setNotificationFilter` will accept `filter` as argument



Test the action creators

Peers(/users/peers)

Import the action types, `NotificationTypeFilters`, and the action creators into

`src/actions/notificationActionCreators.test.js`




Discord(<https://discord.com/app>)



In this file, write a test for the `markAsAread` action. Calling the creator with 1 as an argument should return:



My Profile(/users/my_profile)



```
{
  type: MARK_AS_READ,
  index: 1
}
```


 Home(/)
Write a test for the `setNotificationFilter` action. Calling the creator with one of the filters from `NotificationTypeFilters` as an argument should return:

```
 My Planning(/planning/me)
{
  type: SET_TYPE_FILTER,
  filter: "DEFAULT"
}
 Projects(/projects/current)
```

 QA Reviews I can make(/corrections/to_review)

Repo:

-  • GitHub repository: alx-react
Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)
• Directory: 0x07-react_redux_action_creator_normalizr
• File: task_5/dashboard/src/actions/notificationActionTypes.js,
task_5/dashboard/src/actions/notificationActionCreators.js,
task_5/dashboard/src/actions/notificationActionCreators.test.js


 Concepts(/concepts)
6. Bound the actions

mandatory

 Modify the `Course` actions creators: `Conferences(/dashboards/video_rooms)`


- bound the `selectCourse` action creator
- bound the `unselectCourse` action creator

Modify the `Notification` actions creators:

-  `Sandboxes(/user_containers/current)`
- bound the `markAsRead` action creator
 - bound the `setNotificationFilter` action creator

 Modify the `UI` actions creators: `Tools(/dashboards/my_tools)`

- bound the `login` action creator
- bound the `logout` action creator
- bound the `displayNotificationDrawer` action creator
- bound the `hideNotificationDrawer` action creator

 Peers(/users/peers)
Repo:

- GitHub repository: alx-react
- Discord(<https://discord.com/app>)
- Directory: 0x07-react_redux_action_creator_normalizr
- File: task_6/dashboard/src/actions/courseActionCreators.js,
task_6/dashboard/src/actions/notificationActionCreators.js,
task_6/dashboard/src/actions/uiActionCreators.js

My Profile(/users/my_profile)



7. Async Action Creators

mandatory

Set up Redux and Redux Thunk

Install `redux` and `redux-thunk` in your project

Copy the file `login-success.json` into the `dist` folder. You can do the same with the `notifications.json` file as well now

These files will be available on the web server and will be your own API

Modify the file named `uiActionTypes.js`, add two action types:

- `LOGIN_SUCCESS`
- `LOGIN_FAILURE`

Modify the `uiActionCreators` file:

- Create a `loginSuccess` action creator, that will return the previously created type
- Create a `loginFailure` action creator, that will return the previously created type

Create a `loginRequest` function that takes into argument the `email` and `password` of the user:

- the function should dispatch the `login` action using the action creator previously created
- the function should fetch the API `/login-success.json` and if it succeeds, dispatch the `loginSuccess` action
- if the API fails, dispatch the `loginFailure` action

Write the tests

In the file `uiActionCreators.test.js`, write a test suite for the `loginRequest` action:

- the first test should verify that if the API returns the right response, the store received two actions `LOGIN` and `LOGIN_SUCCESS`
- the first test should verify that if the API query fails, the store received two actions `LOGIN` and `LOGIN_FAILURE`

Tips:

- You can use `jest-fetch-mock` to query an API
- You can install `redux-mock-store` and `fetch-mock` to simulate the API and simulate the store
- With `fetch-mock`, you can use `getOnce` and `get` to simulate success and failures

Requirements:

- All the tests in the project should pass

Discord(<https://discord.com/app>)

Repo:

- GitHub repository: `alx-react`
- Directory: `0x07-react_redux_action_creator_normalizr`
- File: `task_7/dashboard/src/actions/uiActionTypes.js`,
`task_7/dashboard/src/actions/uiActionCreators.js`,





(/)



Home(/)



My Planning(/planning/me)



Projects(/projects/current)

Copyright © 2024 ALX, All rights reserved.



QA Reviews I can make(/corrections/to_review)



Evaluation quizzes(/dashboards/my_current_evaluation_quizzes)



Curriculums(/dashboards/my_curriculums)



Concepts(/concepts)



Conference rooms(/dashboards/video_rooms)



Servers(/servers)



Sandboxes(/user_containers/current)



Tools(/dashboards/my_tools)



Video on demand(/dashboards/videos)



Peers(/users/peers)



Discord(<https://discord.com/app>)



My Profile(/users/my_profile)