Arunava Modak
Posted on Nov 22, 2021 • Updated on Jan 19, 2022

Make A ToDo App using React and ReduxToolkit

#reduxtoolkit #beginners #programming #tutorial

In this article we will learn the very basics of redux toolkit and make a simple app using react and redux toolkit. I have kept it very simple with respect to styling, using a little bit css, but you guys can choose whatever you need for styling.

What Is Redux ?

Redux is an open-source JavaScript library for managing and centralizing application state. It helps us to write JavaScript apps that behave consistently across client, server, and native environments and are easy to test. While it's mostly used as a state management tool with React.

What Is RTK And Why Do We Need It ?

According to the official documentation,

> Redux Toolkit is our official recommended approach for writing Redux logic. It wraps around the Redux core, and contains packages and functions that we think are

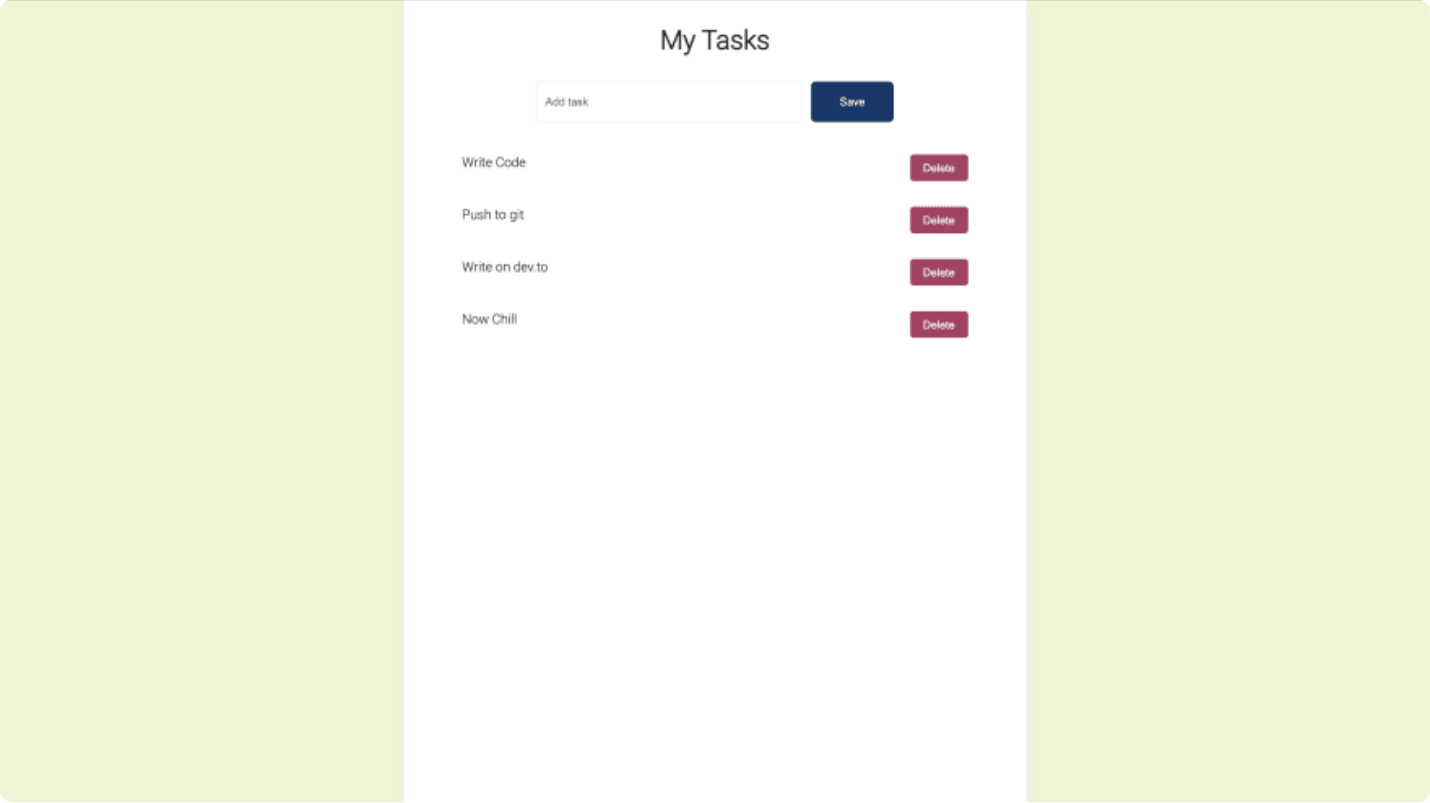
> essential for building a Redux app. Redux Toolkit builds in our suggested best practices, simplifies most Redux tasks, prevents common mistakes, and makes it easier to write Redux applications.

It was originally created to help address three common concerns about Redux:

- "Configuring a Redux store is too complicated"
- "I have to add a lot of packages to get Redux to do anything useful"
- "Redux requires too much boilerplate code"

You can read more about redux toolkit here.

So, lets take a peek at what we are looking to build



Step 1 - Initialize App and Install Dependencies

```
npx create-react-app rtk-todo --use-npm
```

this step initializes a basic react-app for you using npm. Next you need to install the dependencies.

```
npm install react-redux @reduxjs/toolkit
```

now run this following command to start the development server

```
npm start
```

Step 2 - Setup Your Folder Structure

Create a structure like this inside the src of your project, we will create our components in the components folder and put all store related stuff inside the redux folder.

```
📦 src
├ 📂 components
| ├ 📜 AddTodo.js
| ├ 📜 TodoItem.js
| └ 📜 TodoList.js
├ 📂 redux
| ├ 📜 store.js
| └ 📜 tasksSlice.js
├ 📜 App.js
├ 📜 index.css
└ 📜 index.js
```

Step 3 - Redux

Configuring a Redux is an extremely simple process with Redux toolkit. Let's start with writing our first slice.

```
1   import { createSlice } from "@reduxjs/toolkit";
2
3   export const tasksSlice = createSlice({
4       name: "tasks",
5       initialState:[],
6       reducers:{
7           addTask: (state, action)=>{
8               const newTask = {
9                   id: new Date(),
10                  name: action.payload.task
11              }
12              state.push(newTask);
13          },
14          deleteTask: (state, action)=>{
15              return state.filter((item) => item.id !== action.payload.id);
```

```
        }
    }
});

export const {addTask, deleteTask} = tasksSlice.actions;

export default tasksSlice.reducer;
```

rtk-todo-taskSlice.js hosted with ❤ by GitHub view raw

Here we are using `createSlice`, it takes a 'slice name', 'initial state' and an object of reducers, and then generates corresponding action generators and action creators to reducers, and each reducer has access to the state and the action.

Then using this we need to configure out store. For that we need to use `configueStore`. It is an abstraction over the standard Redux createStore function adding some defaults easing our setup process.

```
import { configureStore } from "@reduxjs/toolkit";
import taskReducer from "./tasksSlice";

export default configureStore({
    reducer:{
        tasks: taskReducer,
    }
});
```

rtk-todo-store.js hosted with ❤ by GitHub view raw

Here we pass the slice reducer we created.

Note : If there is one function (like in our case), it is considered as the root reducer. If we have more than one a `rootReducer` is created behind the scenes, using the `combineReducers` functionality.

Next we just use this store in our index.js, like we used to do in normal redux.

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import store from "./redux/store";
import { Provider } from "react-redux";
```

```
7
8   ReactDOM.render(
9
10    <React.StrictMode>
11      <Provider store={store}>
12        <App />
13      </Provider>
14    </React.StrictMode>,
15    document.getElementById('root')
16  );
```

rtk-todo-index.js hosted with ❤ by GitHub view raw

Step 4 - UI Components

Before building any UI, we should always visualize our component tree. In this app too we would follow a structure something like this.

App
├ AddTodo
└ TodoList
└ TodoItem

In App.js we just call our components and wrap them together.

```
1   import React from 'react';
2   import AddTodo from './components/AddTodo';
3   import TodoList from './components/TodoList';
4
5   const App = () => {
6     return (
7       <div className="app">
8         <h1 className="app-title">My Tasks</h1>
9         <AddTodo />
10        <TodoList />
11      </div>
12    );
13  };
14
15  export default App;
```

rtk-todo-App.js hosted with ❤ by GitHub view raw

Next, we make a way to enter the task and dispatch the `addTask` action.

```jsx
import React, { useState } from 'react';
import { useDispatch } from "react-redux";
import { addTask } from "../redux/tasksSlice";

const AddTodo = () => {
  const [value, setValue] = useState('');

  const dispatch = useDispatch();

  const onSubmit = (event) => {
    event.preventDefault();

    if(value.trim().length === 0)
    {
      alert("Enter a task before adding !!");
      setValue("");
      return;
    }

    dispatch(
      addTask({
        task: value
      })
    );

    setValue("");
  };

  return (
    <div className="add-todo">
      <input
        type="text"
        className="task-input"
        placeholder="Add task"
        value={value}
        onChange={(event) => setValue(event.target.value)}
      ></input>

      <button className="task-button" onClick={onSubmit}>
        Save
      </button>
    </div>
  );
};
```

```
45
46  export default AddTodo;
```

rtk-todo-AddTodo.js hosted with ❤️ by GitHub view raw

We can easily import the action from the slice and dispatch it using the `useDispatch` function.

Now that we can add the task we need to display them. For that we can just access the state using `useSelector` and map over the todo list to display as we want.

```
1   import React from 'react';
2   import TodoItem from './TodoItem';
3   import { useSelector } from "react-redux";
4
5   const TodoList = () => {
6     const todos = useSelector((state)=>{
7       return state.tasks;
8     });
9
10    return (
11      <ul className="tasks-list">
12        {todos.map((todo) => (
13          <TodoItem id={todo.id} title={todo.name} completed={todo.status} />
14        ))}
15      </ul>
16    );
17  };
18
19  export default TodoList;
```

rtk-todo-TodoList.js hosted with ❤️ by GitHub view raw

Almost done, we just need to display each task and call the `deleteTask` action just like we had called the `addTask` action.

```
1   import React from 'react';
2   import { useDispatch } from "react-redux";
3   import { deleteTask } from "../redux/tasksSlice";
4
5   const TodoItem = ({ id, title }) => {
6
7     const dispatch = useDispatch();
8
```

```
 9     const removeTask=()=>{
10       dispatch(
11         deleteTask({
12           id: id
13         })
14       )
15     }
16
17     return (
18       <li className="task-item">
19         <div>
20           {title}
21         </div>
22         <div>
23           <button className="remove-task-button" onClick={()=>{
24             removeTask();
25           }}>Delete</button>
26         </div>
27       </li>
28     );
29   };
30
31   export default TodoItem;
```

rtk-todo-TodoItem.js hosted with ❤️ by GitHub view raw

So now if we try to check the progress, we can see that we can add or delete any task and it looks like this, and its quite bad.

So now we add the styles, for this tutorial I have only added basic CSS, but you guys can go wild, maybe use a UI framework like Antd or MaterialUI too, would surely look nice.

```
body{
    background-color: #edf5d1;
    padding: 0px;
    margin: 0px;
    color: rgba(0,0,0,0.87);
    font-family: 'Roboto', sans-serif;
}

.app{
    text-align: center;
    background-color: white;
```

```
12      height: auto;
13      min-height: 100vh;
14      width : 40%;
15      margin: auto;
16      padding : 30px;
17      box-shadow: 2px 2px 20px #ececec;
18  }
19
20  .app-title{
21      margin: 0;
22      margin-bottom: 30px;
23  }
24
25  .add-todo{
26      margin: 30px 0px;
27  }
28
29  .task-input{
30      height : 45px;
31      width : 300px;
32      border : 1px solid #e4f2f7;
33      border-radius: 6px;
34      padding : 2px 10px;
35      margin-right : 10px;
36  }
37
38  .task-input:focus {
39      box-shadow: 0 0 8px rgba(142,228,175,0.6);
40      outline: none;
41  }
42
43  .task-button{
44      height : 49px;
45      width : 100px;
46      border : none;
47      border-radius: 6px;
48      background-color: #05386b;
49      color:white;
50  }
51
52  .task-button:hover{
53      cursor: pointer;
54  }
55
```

```css
.tasks-list{
    list-style: none;
    margin: auto;
    padding : 0px;
}

.task-item{
    margin: 15px 0px;
    display: flex;
    justify-content: space-between;
    padding : 8px 40px;
}

.remove-task-button{
    background-color: #ac3b61;
    height: 32px;
    color: white;
    border-radius: 4px;
    width: 70px;
    border:none;
}

.remove-task-button:hover{
    cursor: pointer;
}
```

rtk-todo-index.css hosted with ❤ by GitHub view raw

And thats done, it looks excatly like we aimed it to be.

Untill next time Happy Building !! 😎

Top comments (2)

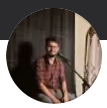
ayomiku olatunji • Feb 24 '22

Delete not working

Arunava Modak • Nov 22 '21

In case you want to compare codes while building this, here is the code for this, github.com/arunavamodak/redux_tool.... 😊

Code of Conduct • Report abuse



Arunava Modak

A Software Engineer, in love with building things. Passionate, especially about beautiful UI.

LOCATION
Bengaluru, India

WORK
Software Engineer @ OLX Group

JOINED
Nov 12, 2021

Trending on DEV Community 🔥


How to slugify a string in JavaScript
#javascript #webdev #beginners #node


CodeNewbies! Help Us Create Your Community
#discuss #beginners #codenewbie


How Do You Make Lifelong Learning a Habit?
#discuss #beginners #codenewbie #career