Redux Toolkit

React Course

My React Course

Support

Find the App Useful? You can always buy me a coffee

Docs

Redux Toolkit Docs

Install Template

```
npx create-react-app my-app --template redux
```

- @latest

```
npx create-react-app@latest my-app --template redux
```

Existing App

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

@reduxjs/toolkit

consists of few libraries

- redux (core library, state management)
- immer (allows to mutate state)
- redux-thunk (handles async actions)
- reselect (simplifies reducer functions)

Extras

- redux devtools
- combine reducers

react-redux

connects our app to redux

Setup Store

- create store.js

```js
import { configureStore } from '@reduxjs/toolkit';

export const store = configureStore({
  reducer: {},
});
```

Setup Provider

- index.js

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

ReactDOM.render(
  <React.StrictMode>
    <Provider store={store}>
      <App />
    </Provider>
  </React.StrictMode>,
  document.getElementById('root')
);
```

Setup Cart Slice

- application feature
- create features folder/cart
- create cartSlice.js

```js
import { createSlice } from '@reduxjs/toolkit';

const initialState = {
  cartItems: [],
  amount: 0,
  total: 0,
  isLoading: true,
};
```

```
const cartSlice = createSlice({
  name: 'cart',
  initialState,
});

console.log(cartSlice);

export default cartSlice.reducer;
```

- store.js

```
import { configureStore } from '@reduxjs/toolkit';
import cartReducer from './features/cart/cartSlice';

export const store = configureStore({
  reducer: {
    cart: cartReducer,
  },
});
```

Redux DevTools

- extension

Access store value

- create components/Navbar.js

```
import { CartIcon } from '../icons';
import { useSelector } from 'react-redux';

const Navbar = () => {
  const { amount } = useSelector((state) => state.cart);

  return (
    <nav>
      <div className='nav-center'>
        <h3>redux toolkit</h3>
        <div className='nav-container'>
          <CartIcon />
          <div className='amount-container'>
            <p className='total-amount'>{amount}</p>
          </div>
        </div>
      </div>
    </nav>
  );
};
export default Navbar;
```

Hero Icons

- Hero Icons

```
nav svg {
  width: 40px;
  color: var(--clr-white);
}
```

Setup Cart

- cartSlice.js

```
import cartItems from '../../cartItems';

const initialState = {
  cartItems: cartItems,
  amount: 0,
  total: 0,
  isLoading: true,
};
```

- create CartContainer.js and CartItem.js
- CartContainer.js

```
import React from 'react';
import CartItem from './CartItem';
import { useSelector } from 'react-redux';

const CartContainer = () => {
  const { cartItems, total, amount } = useSelector((state) => state.cart);

  if (amount < 1) {
    return (
      <section className='cart'>
        {/* cart header */}
        <header>
          <h2>your bag</h2>
          <h4 className='empty-cart'>is currently empty</h4>
        </header>
      </section>
    );
  }
  return (
    <section className='cart'>
      {/* cart header */}
      <header>
        <h2>your bag</h2>
      </header>
      {/* cart items */}
      <div>
        {cartItems.map((item) => {
          return <CartItem key={item.id} {...item} />;
        })}
      </div>
      {/* cart footer */}
      <footer>
        <hr />
        <div className='cart-total'>
          <h4>
            total <span>${total}</span>
          </h4>
        </div>
        <button className='btn clear-btn'>clear cart</button>
      </footer>
    </section>
  );
};

export default CartContainer;
```

- CartItems.js

```
import React from 'react';
import { ChevronDown, ChevronUp } from '../icons';

const CartItem = ({ id, img, title, price, amount }) => {
  return (
    <article className='cart-item'>
      <img src={img} alt={title} />
      <div>
        <h4>{title}</h4>
        <h4 className='item-price'>${price}</h4>
        {/* remove button */}
        <button className='remove-btn'>remove</button>
      </div>
      <div>
        {/* increase amount */}
        <button className='amount-btn'>
          <ChevronUp />
        </button>
        {/* amount */}
        <p className='amount'>{amount}</p>
        {/* decrease amount */}
        <button className='amount-btn'>
```

```
          <ChevronDown />
        </button>
      </div>
    </article>
  );
};

export default CartItem;
```

First Reducer

- cartSlice.js
- Immer library

```
const cartSlice = createSlice({
  name: 'cart',
  initialState,
  reducers: {
    clearCart: (state) => {
      state.cartItems = [];
    },
  },
});

export const { clearCart } = cartSlice.actions;
```

- create action

```
const ACTION_TYPE = 'ACTION_TYPE';

const actionCreator = (payload) => {
  return { type: ACTION_TYPE, payload: payload };
};
```

- CartContainer.js

```
import React from 'react';
import CartItem from './CartItem';
import { useDispatch, useSelector } from 'react-redux';

const CartContainer = () => {
  const dispatch = useDispatch();

  return (
    <button
      className='btn clear-btn'
      onClick={() => {
        dispatch(clearCart());
      }}
    >
      clear cart
    </button>
  );
};

export default CartContainer;
```

Remove, Increase, Decrease

- cartSlice.js

```
import { createSlice } from '@reduxjs/toolkit';
import cartItems from '../../cartItems';

const initialState = {
  cartItems: [],
  amount: 0,
  total: 0,
  isLoading: true,
};

const cartSlice = createSlice({
  name: 'cart',
  initialState,
  reducers: {
    clearCart: (state) => {
      state.cartItems = [];
    },
    removeItem: (state, action) => {
      const itemId = action.payload;
      state.cartItems = state.cartItems.filter((item) => item.id !== itemId);
    },
    increase: (state, { payload }) => {
      const cartItem = state.cartItems.find((item) => item.id === payload.id);
      cartItem.amount = cartItem.amount + 1;
    },
    decrease: (state, { payload }) => {
      const cartItem = state.cartItems.find((item) => item.id === payload.id);
      cartItem.amount = cartItem.amount - 1;
    },
    calculateTotals: (state) => {
      let amount = 0;
      let total = 0;
      state.cartItems.forEach((item) => {
        amount += item.amount;
        total += item.amount * item.price;
      });
      state.amount = amount;
      state.total = total;
    },
  },
});

export const { clearCart, removeItem, increase, decrease, calculateTotals } =
  cartSlice.actions;

export default cartSlice.reducer;
```

- CartItem.js

```
import React from 'react';
import { ChevronDown, ChevronUp } from '../icons';

import { useDispatch } from 'react-redux';
import { removeItem, increase, decrease } from '../features/cart/cartSlice';

const CartItem = ({ id, img, title, price, amount }) => {
  const dispatch = useDispatch();

  return (
    <article className='cart-item'>
      <img src={img} alt={title} />
      <div>
        <h4>{title}</h4>
        <h4 className='item-price'>${price}</h4>
        {/* remove button */}
        <button
          className='remove-btn'
          onClick={() => {
            dispatch(removeItem(id));
          }}
        >
          remove
        </button>
      </div>
      <div>
        {/* increase amount */}
        <button
          className='amount-btn'
          onClick={() => {
            dispatch(increase({ id }));
          }}
        >
          <ChevronUp />
        </button>
        {/* amount */}
        <p className='amount'>{amount}</p>
        {/* decrease amount */}
        <button
          className='amount-btn'
          onClick={() => {
            if (amount === 1) {
              dispatch(removeItem(id));
              return;
            }
            dispatch(decrease({ id }));
          }}
        >
          <ChevronDown />
        </button>
      </div>
    </article>
  );
};

export default CartItem;
```

- App.js

```
import { useEffect } from 'react';
import Navbar from './components/Navbar';
import CartContainer from './components/CartContainer';
import { useSelector, useDispatch } from 'react-redux';
import { calculateTotals } from './features/cart/cartSlice';

function App() {
  const { cartItems } = useSelector((state) => state.cart);
  const dispatch = useDispatch();
  useEffect(() => {
    dispatch(calculateTotals());
  }, [cartItems]);

  return (
    <main>
      <Navbar />
      <CartContainer />
    </main>
  );
}

export default App;
```

Modal

- create components/Modal.js

```
const Modal = () => {
  return (
    <aside className='modal-container'>
```

```
        <div className='modal'>
          <h4>Remove all items from your shopping cart?</h4>
          <div className='btn-container'>
            <button type='button' className='btn confirm-btn'>
              confirm
            </button>
            <button type='button' className='btn clear-btn'>
              cancel
            </button>
          </div>
        </div>
      </aside>
    );
  };
  export default Modal;
```

- App.js

```
return (
  <main>
    <Modal />
    <Navbar />
    <CartContainer />
  </main>
);
```

modal slice

- create features/modal/modalSlice.js

```
import { createSlice } from '@reduxjs/toolkit';
const initialState = {
  isOpen: false,
};

const modalSlice = createSlice({
  name: 'modal',
  initialState,
  reducers: {
    openModal: (state, action) => {
      state.isOpen = true;
    },
    closeModal: (state, action) => {
      state.isOpen = false;
    },
  },
});
```

```
export const { openModal, closeModal } = modalSlice.actions;
export default modalSlice.reducer;
```

- App.js

```
const { isOpen } = useSelector((state) => state.modal);

return (
  <main>
    {isOpen && <Modal />}
    <Navbar />
    <CartContainer />
  </main>
);
```

toggle modal

- CartContainer.js

```
import { openModal } from '../features/modal/modalSlice';

return (
  <button
    className='btn clear-btn'
    onClick={() => {
      dispatch(openModal());
    }}
  >
    clear cart
  </button>
);
```

- Modal.js

```
import { closeModal } from '../features/modal/modalSlice';
import { useDispatch } from 'react-redux';
import { clearCart } from '../features/cart/cartSlice';

const Modal = () => {
  const dispatch = useDispatch();

  return (
    <aside className='modal-container'>
      <div className='modal'>
        <h4>Remove all items from your shopping cart?</h4>
        <div className='btn-container'>
          <button
```

```
            type='button'
            className='btn confirm-btn'
            onClick={() => {
              dispatch(clearCart());
              dispatch(closeModal());
            }}
          >
            confirm
          </button>
          <button
            type='button'
            className='btn clear-btn'
            onClick={() => {
              dispatch(closeModal());
            }}
          >
            cancel
          </button>
        </div>
      </div>
    </aside>
  );
};
export default Modal;
```

async functionality with createAsyncThunk

- Course API
- https://course-api.com/react-useReducer-cart-project
- cartSlice.js
- action type
- callback function
- lifecycle actions

```
import { createSlice, createAsyncThunk } from '@reduxjs/toolkit';

const url = 'https://course-api.com/react-useReducer-cart-project';

export const getCartItems = createAsyncThunk('cart/getCartItems', () => {
  return fetch(url)
    .then((resp) => resp.json())
    .catch((err) => console.log(error));
});

const cartSlice = createSlice({
  name: 'cart',
  initialState,
  extraReducers: {
    [getCartItems.pending]: (state) => {
      state.isLoading = true;
    },
    [getCartItems.fulfilled]: (state, action) => {
      console.log(action);
      state.isLoading = false;
      state.cartItems = action.payload;
    },
    [getCartItems.rejected]: (state) => {
      state.isLoading = false;
    },
  },
});
```

- App.js

```
import { calculateTotals, getCartItems } from './features/cart/cartSlice';

function App() {
  const { cartItems, isLoading } = useSelector((state) => state.cart);

  useEffect(() => {
    dispatch(getCartItems());
  }, []);

  if (isLoading) {
    return (
      <div className='loading'>
        <h1>Loading...</h1>
      </div>
    );
  }

  return (
    <main>
      {isOpen && <Modal />}
      <Navbar />
      <CartContainer />
    </main>
  );
}

export default App;
```

Options

```
npm install axios
```

- cartSlice.js

```js
export const getCartItems = createAsyncThunk(
  'cart/getCartItems',
  async (name, thunkAPI) => {
    try {
      // console.log(name);
      // console.log(thunkAPI);
      // console.log(thunkAPI.getState());
      // thunkAPI.dispatch(openModal());
      const resp = await axios(url);

      return resp.data;
    } catch (error) {
      return thunkAPI.rejectWithValue('something went wrong');
    }
  }
);
```