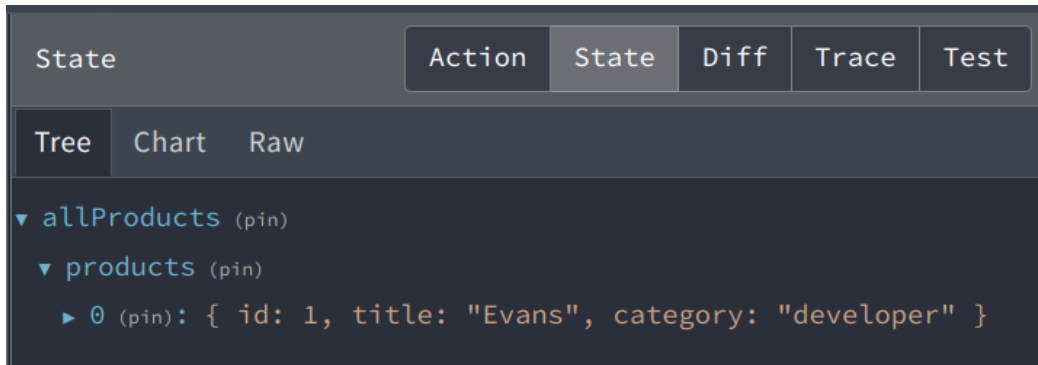


=> reducer always takes initial state and action (state,action)

=> now we will link our store with our react app

-- now you can see that our store state in redux tab



-- and our store contains only one product for now

?? and if want to access the state < item > in any component how we can do that ?

->

in case you struggling with react-router-dom, they had updates, so you need to do it like that :

```
<div className="App">
  <BrowserRouter>
    <Header />
    <Routes>
      <Route path="/productlist" element={<ProductList />}/>
    </Routes>
  </BrowserRouter>
</div>
```

Show less

@zakazaka2957 replace <switch></switch> with <Routes></Routes> and replace the import too

=> to access the redux store we use useSelector hook

-- provided by the redux

```
const ProductListing = () => {
  const products = useSelector((state) => state);
  console.log(products);
  return (
    <div className='ui grid container'>
      <h1> ProductListing </h1>
    </div>
  );
}
```

--

@rakibulhasanasif2120 1 year ago

In 33:38 min, Don't get confused guys. just change,
This line,

```
const products = useSelector((state) => state);  
to  
const products = useSelector((state) => state.allProducts.products);  
Thank me later. Maybe somehow that part was cut.
```

Show less

```
▼ Object 1  
  ▼ allProducts:  
    ▼ products: Array(1)  
      ► 0: {id: 1, title: 'Evans', category:  
        length: 1  
      ► [[Prototype]]: Array(0)  
      ► [[Prototype]]: Object  
      ► [[Prototype]]: Object
```

-- and product is coming from our redux,
-- and now we can use this product to display on our app

```
const ProductComponent = () => {  
  const products = useSelector((state) => state.allProducts.products);  
  const {id,title} = products[0];  
  
  return (  
    <div className='four column wide'>  
      { /* <h1> ProductComponent </h1> */ }  
      <div className='ui link cards'>  
        <div className='card'>  
          <div className='image'></div>  
          <div className='content'>  
            <div className='header'>{title}</div>  
          </div>  
        </div>  
      </div>  
    </div>  
  )  
}  
  
export default ProductComponent;
```

FuckShop

Evans

-- and we will be using the fake api store for getting the products / dummy data
-> first we need to setup the axios

Get all products

```
fetch('https://fakestoreapi.com/products')  
  .then(res=>res.json())  
  .then(json=>console.log(json))
```

- : axios to make http calls :
- > so we will be make request at this end point
- > so first we will be importing in the productList component
- > and as we know we call our fetch function inside the useEffect hook

```
const ProductListing = () => {

  const products = useSelector((state) => state);
  // console.log(products);
  const fetchProducts = async () => {
    const response = await axios.get("https://fakestoreapi.com/products")
    .catch((err) => {
      console.log('Err', err);
    });
    console.log(response);
  };

  useEffect(() => {
    fetchProducts();
  }, [])
}
```

- > and here you can see that we are getting the result

Name	✕ Headers Preview Response Initiator Timing
products	▼ [{id: 1, title: "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Lapt
products	▶ 0: {id: 1, title: "Fjallraven - Foldsack No. 1 Backpack, Fits 15 L ▶ 1: {id: 2, title: "Mens Casual Premium Slim Fit T-Shirts ", price: ▶ 2: {id: 3, title: "Mens Cotton Jacket", price: 55.99,...} ▶ 3: {id: 4, title: "Mens Casual Slim Fit", price: 15.99,...} ▶ 4: {id: 5, title: "John Hardy Women's Legends Naga Gold & Silver D ▶ 5: {id: 6, title: "Solid Gold Petite Micropave ", price: 168,...} ▶ 6: {id: 7, title: "White Gold Plated Princess", price: 9.99,...} ▶ 7: {id: 8, title: "Pierced Owl Rose Gold Plated Stainless Steel Do ▶ 8: {id: 9, title: "WD 2TB Elements Portable External Hard Drive - I ▶ 9: {id: 10, title: "SanDisk SSD PLUS 1TB Internal SSD - SATA III 6 ▶ 10: {id: 11, title: "Silicon Power 256GB SSD 3D NAND A55 SLC Cache

- > so our api is working and now we have to display these things on the app

=> since we can see that all the products are in the data part so we can do
=> response.data

```
(20) [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}]
▼ {...} ⓘ
  ▶ 0: {id: 1, title: 'Fjallraven - Foldsack No. 1 Backpack,
  ▶ 1: {id: 2, title: 'Mens Casual Premium Slim Fit T-Shirts
  ▶ 2: {id: 3, title: 'Mens Cotton Jacket', price: 55.99, de
  ▶ 3: {id: 4, title: 'Mens Casual Slim Fit', price: 15.99
```

=> since we are getting this data from the server so we need to store this data in our redux store

? how to add server response to the redux store ?
=> let's see

=> so we need to dispatch an action and it is setProduct :

=> so our actions have setProduct

```
export const setProducts = (products) => {
  return {
    type : Actiontypes.SET_PRODUCTS,
    payload : products,
  };
};
```

-- which takes the products

=> to dispatch we have a redux hook called useDispatch

```
const products = useSelector((state) => state.products);
const dispatch = useDispatch();

// console.log(response.data);
dispatch(setProducts(response.data));
```

=> and it returns this object and it will be taken by reducer

```
export const productReducer = (state = initialState, {type, payload}) => {
  switch(type) {
    case Actiontypes.SET_PRODUCTS :
      return state;

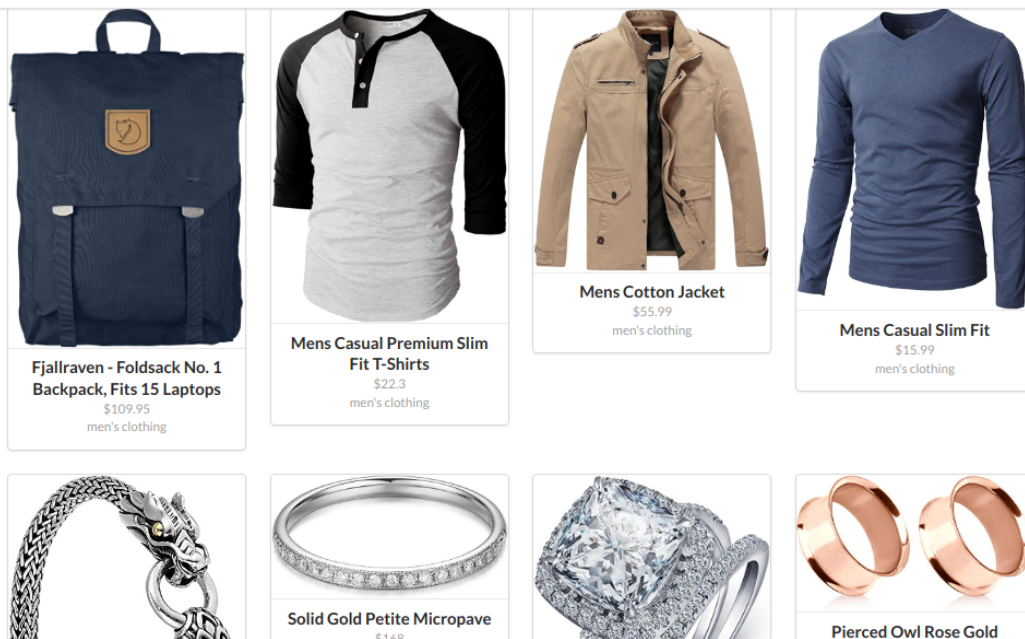
    default :
      return state;
  }
}
```

-- and here it accepts the type and payload

-- since we are getting the array so we need to map

```
const ProductComponent = () => {
  const products = useSelector((state) => state.allProducts.products);
  const renderList = products.map((product) => {
    const {id, title, image, price, category} = product;
    return (
      <div className='four column wide' key={id}>
        { /* <h1> ProductComponent </h1> */ }
        <div className='ui link cards'>
          <div className='card'>
            <div className='image'>
              <img src={image} alt={title} />
            </div>
            <div className='content'>
              <div className='header'>{title}</div>
              <div className='meta price'>${price}</div>
              <div className='meta'>{category}</div>
            </div>
          </div>
        </div>
      </div>
    );
  });
};
```

FuckShop



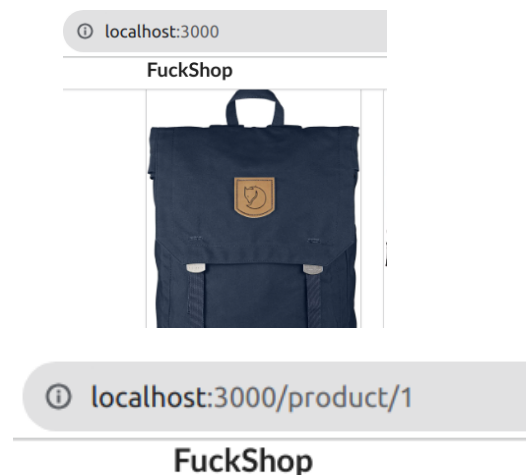
- now you can see that we are getting the products and now

==> we should go on each individual product when i click on them :
 ==> so we will using Link which helps us to navigate to product details

```
> containers > JS ProductComponent.js > ...
1 import React from 'react'
2 import { useSelector } from 'react-redux';
3 import { Link } from 'react-router-dom';
```

==> so we need to wrap our component inside the Link tag
 ==> and as you can see that we are click we are going on product page /address

```
<Link to={` /product/${id}`}>
  { /* <h1> ProductComponent </h1> */ }
  <div className="ui link cards">
    <div className='card'>
      <div className='image'>
        <img src={image} alt={title} />
      </div>
      <div className='content'>
        <div className='header'>{title}</div>
        <div className='meta price'>${price}</div>
        <div className='meta'>{category}</div>
      </div>
    </div>
  </div>
</Link>
```



==> and since we need this product parameter : product id and again use the fake api store , to make a call and use api for the single product

Get a single product

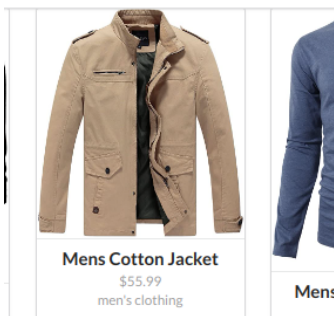
```
fetch('https://fakestoreapi.com/products/1')
  .then(res=>res.json())
  .then(json=>console.log(json))
```

-- and at product details again we will be using axios to make the api calls
=> and we need to get the param values so we need to use useParams
-- useParams helps to get the parameter value

```
import React, {useEffect} from 'react'
import axios from 'axios'
import { useParams } from 'react-router-dom'
```

```
const ProductDetail = () => {
  const {productId} = useParams();
  console.log(productId);
}
```

-- and now you can see that when we are clicking on the item we are getting the product Id



Products [ProductListing.js:25](#)
▶ {allProducts: {...}}
3 [ProductDetail.js:8](#)
3 [ProductDetail.js:8](#)

-- and using this id we can make an api call : and display the details of the product :
-- and to access the selectedProduct we need to use selector as previous
=> and we need reducer for this so we will be creating the reducer

```
const product = useSelector((state) => state.product)
```

--

```
export const selectedProductReducer = (state={}, {type, payload}) => {
  switch(type) {
    case ActionTypes.SELECTED_PRODUCT:
      return {...state, ...payload};
    default:
      state;
  }
}
```

-- so now we have reducer and let's go to product details and before that we need to add this reducer to the combined reducer
=> let's see the product :

=> now you can see that when we are clicking on that we are getting the product details



```
ProductDetail.js:13
{id: 4, title: 'Mens Casual Slim Fit', price: 15.99, description: 'The color could be slightly different between on t...uld be reviewed below on the product description.', category: "men's clothing", ...}
```

-- and now we need to display the data < via writing the jsx >

=> so finally it's working and

** there is a problem whenever i am clicking on the product first it displays the old one and then new one : so we need to remove the product from the selected when it get destroyed :

=> and we have our action type remove selected product :

```
export const ActionTypes = {
  SET_PRODUCTS: "SET_PRODUCTS",
  SELECTED_PRODUCT: "SELECTED_PRODUCT",
  REMOVE_SELECTED_PRODUCT: "REMOVE_SELECTED_PRODUCT",
};
```

=> let's create the action :

```
export const removeSelectedProduct = (product) => {
  return {
    type: ActionTypes.SELECTED_PRODUCT,
    // payload: product,
    // we didnt get anything
  };
};
```

-- and now go to the reducer and add the condition for the reducer

```
case ActionTypes.SELECTED_PRODUCT:
  return {...state, ...payload};
```

```
case ActionTypes.REMOVE_SELECTED_PRODUCT:
  ...return{};
```

=> added one more condition so when remove selected then return empty :

=> and now we can go to the details and dispatch the action

```
useEffect(() => {
  if (productId && productId !== "") fetchProductDetail();
  return () => {
    dispatch(removeSelectedProduct)
  }
});
```