Redux

# 下载 redux

1 npm i redux

1. 配置仓库

数据初始化：

* 1. 创建⼀个存储数据的仓库

1. import { createStore } from "redux";
2. const store = createStore();
   1. 设置初始数据

Redux 中的 state 统⼀交给 reducer 处理。

1. export const count = (state = 0, action) => {
2. return state; 3 }
   1. 将数据添加到仓库
3. import { createStore, combineReducers } from "redux";
4. import { count } from "./counter/reducers.js";
5. const store = createStore(
6. combineReducers({ count }) 5 );
7. 连接组件与仓库
   1. 下载 react-redux

1 npm i react-redux

* 1. 连接 store 与组件

import { Provider } from "react-redux";

import store from "./store/store.js"

1

2

3

4

5

6

7

8

9

10

* 1. 组件访问仓库

class Counter extends Component {

render() {

return (

<>

<h1>计数器：{this.props.count}</h1>

</>

)

}

}

// state 获取到的是仓库中所有数据

const mapStateToProps = state => { return { count: state.count };

}

export default connect(mapStateToProps)(Counter);

import React, { Component } from 'react'

import { connect } from "react-redux";

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

export default function App() {

return (

<Provider store={store}>

<RootRouter></RootRouter>

</Provider>

)

}

* 1. redux-actions

1 npm i redux-actions

* + 1. state 在 store 内
    2. 通过 reducer 修改 state（reducer 函数的返回值会覆盖掉 store 中的旧state）
    3. 通过 dispatch 触发 reducer 的执⾏
       1. 调⽤ dispatch 时传递⼀个 action 对象：{ type: '任务⼀'} 拆分：

1. reducers.js：所有任务对应的操作
2. actions.js：{ type: '任务⼀'}、{ type: '任务⼆'}、{ type: '任务三'} .....
3. actions-type.js：'任务⼀'、'任务⼆' ....