React进阶(Redux)

1. Redux

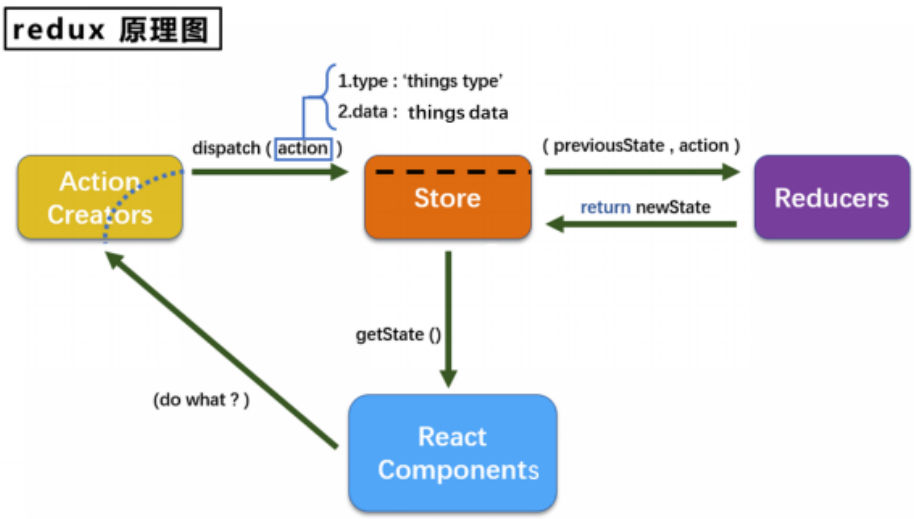
1. Redux是什么?

1. redux是一个专门用于做**状态管理**的JS库(不是react插件库)。
2. 它可以用在react, angular, vue等项目中, 但基本与react配合使用。
3. 作用: 集中式管理react应用中多个组件**共享**的状态。

2. 什么情况下需要使用Redux?

1. 你希望状态有一个唯一确定的来源。
2. 某个组件的状态，需要让其他组件可以随时拿到（共享）。
3. 一个组件需要改变另一个组件的状态（通信）。
4. 你发现将所有状态放在顶层组件中管理已不可维护。
5. 如果你不确定你是否需要它，你可能不需要。

3. Redux工作流程



4. Redux的是三个核心概念

1. action

1. 动作的对象
2. 包含2个属性

- type：标识属性，值为字符串，唯一，必要属性

- data：数据属性，值类型任意，可选属性

3. 例子：{ type: 'ADD_STUDENT', data: {name: 'tom',age:18} }

2. reducer

1. 用于初始化状态、加工状态。
2. 加工时，根据旧的state和action， 产生新的state的**纯函数**。

1. store

1. 将state、action、reducer联系在一起的对象
2. 如何得到此对象?
    a. import {createStore} from 'redux'
    b. import reducer from './reducers'
    c. const store = createStore(reducer)
3. 此对象的功能?
    a. getState(): 得到state
    b. dispatch(action): 分发action，触发reducer调用，产生新的state
    c. subscribe(listener): 注册监听，当产生了新的state时，自动调用

5. redux的核心API

1. createstore()

1. 作用：创建包含指定reducer的store对象

2. store对象

1. 作用：创建包含指定reducer的store对象
2. 它内部维护着:
    a. state
    b. reducer
3. 核心方法:
    a. getState()
    b. dispatch(action)
    c. subscribe(listener)
4. 具体编码:
    a. store.getState()
    b. store.dispatch({type:'INCREMENT', number})
    c. store.subscribe(render)

3. applyMiddleware()

1. 作用：应用上基于redux的中间件(插件库)

4. combineReducers()

1. 作用：合并多个reducer函数

6. redux异步编程

1. 理解

1. redux默认是不能进行异步处理的
2. 某些时候应用中需要在**redux中执行异步任务**(ajax, 定时器)

2. 使用异步中间件

npm install --save redux-thunk

7. react-redux

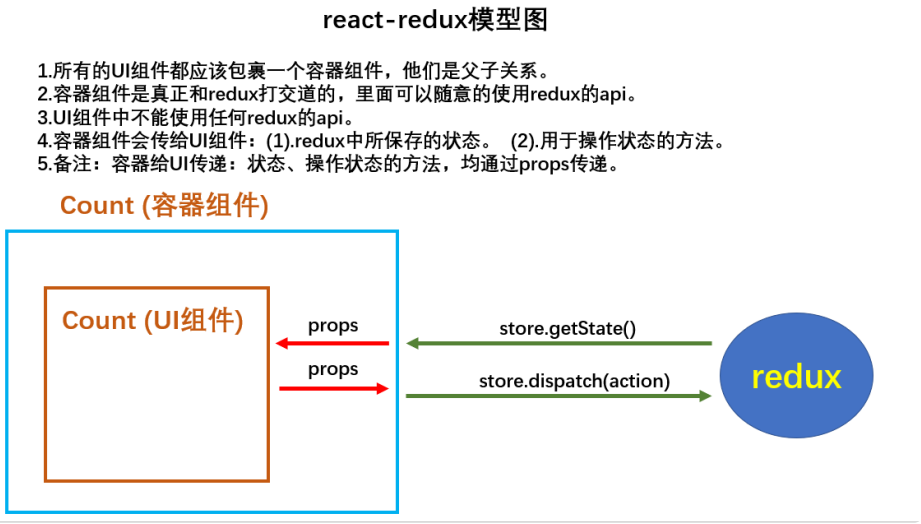
1. 理解

1. 一个react插件库
2. 专门用来简化react应用中使用redux

2. react-Redux组件分类

1. UI组件
    a. 只负责 UI 的呈现，不带有任何业务逻辑
    b. 通过props接收数据(一般数据和函数)
    c. 不使用任何 Redux 的 API
    d. 一般保存在components文件夹下
2. 容器组件
    a. 负责管理数据和业务逻辑，不负责UI的呈现
    b. 使用 Redux 的 API
    c. 一般保存在containers文件夹下

3. react-Redux模型图



4. react-Redux相关API

1. Provider：让所有组件都可以得到state数据

```jsx
<Provider store={store}>
  <App />
</Provider>
```

2. connect：用于包装 UI 组件生成容器组件

```jsx
import { connect } from 'react-redux'

connect(
  mapStateToprops,
  mapDispatchToProps
)(Counter)
```

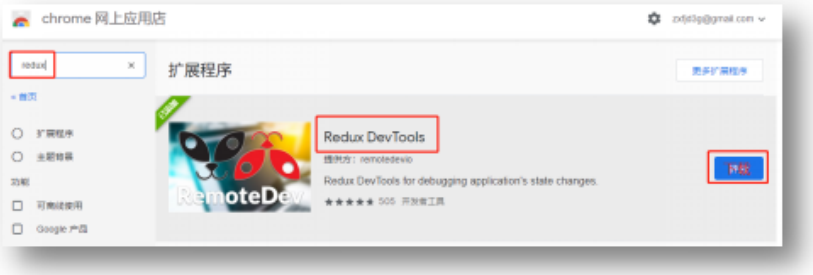
3. mapStateToprops：将外部的数据（即state对象）转换为UI组件的标签属性

```jsx
const mapStateToprops = function (state) {
  return {
    value: state
  }
}
```

4. mapDispatchToProps：将分发action的函数转换为UI组件的标签属性

8. 使用上redux调试工具

1. 安装chrome浏览器插件



2. 下载工具依赖包

npm install --save-dev redux-devtools-extension

9. 纯函数和高阶函数

1. 纯函数

1. 一类特别的函数: 只要是同样的输入(实参)，必定得到同样的输出(返回)
2. 必须遵守以下一些约束
    a. 不得改写参数数据
    b. 不会产生任何副作用，例如网络请求，输入和输出设备
    c. 不能调用Date.now()或者Math.random()等不纯的方法
3. redux的reducer函数必须是一个纯函数

2. 高阶函数

1. 一类特别的函数
    a. 情况1: 参数是函数
    b. 情况2: 返回是函数
2. 常见的高阶函数:
    a. 定时器设置函数
    b. 数组的forEach(), map(), filter(), reduce(), find(), bind()
    c. promise
    d. react-redux中的connect函数
3. 作用: 能实现更加动态, 更加可扩展的功能

参考链接:

React官方文档

尚硅谷2021版React技术全家桶全套完整版 – bilibili

Redux 中文官网

Redux 中文文档

react-redux中文文档

你可能不需要Redux – 简书

Redux中的reducer到底是什么，以及它为什么叫reducer? – 知乎