React-6

上节课内容总结

- react-router

课程目标

- 掌握 redux 三大原则
- 掌握 redux 基础使用
- 掌握 react-redux 使用
- 掌握 redux-thunk 使用

redux

- Redux 是一个独立的 JavaScript 状态管理库，与非 React 内容之一
- https://www.redux.org.cn/

课程内容

为什么使用 redux

React 本身 MVVM 渐进式框架(M(model 数据模型) - V(view 视图) - VM(虚拟模型))

核心概念

理解 Redux 核心几个概念与它们之间的关系

- state
- reducer
- store
- action

state 对象

通常我们会把应用中的数据存储到一个对象树（Object Tree） 中进行统一管理，我们把这个对象树称为：state

state 是只读的

这里需要注意的是，为了保证数据状态的可维护和测试，不推荐直接修改 state 中的原数据

通过纯函数修改 state

什么是纯函数？

纯函数

1. 相同的输入永远返回相同的输出
2. 不修改函数的输入值
3. 不依赖外部环境状态
4. 无任何副作用

使用纯函数的好处

1. 便于测试
2. 有利重构

action 对象

我们对 state 的修改是通过 reducer 纯函数来进行的，同时通过传入的 action 来执行具体的操作，action 是一个对象

- type 属性 : 表示要进行操作的动作类型，增删改查.....
- payload属性 : 操作 state 的同时传入的数据

但是这里需要注意的是，我们不直接去调用 Reducer 函数，而是通过 Store 对象提供的 dispatch 方法来调用

Store 对象

为了对 state，Reducer，action 进行统一管理和维护，我们需要创建一个 Store 对象

redux 三大原则

- 单一数据源: 整个应用的 state 被储存在一棵 object tree 中，并且这个 object tree 只存在于唯一一个 store 中
- State 是只读的: 唯一改变 state 的方法就是触发 action，action 是一个用于描述已发生事件的普通对象
- 使用纯函数来执行修改

redux API

- createStore(reducer, [preloadedState], enhancer);
    - reducer (Function): 接收两个参数，分别是当前的 state 树和要处理的 action，返回新的 state 树。
    - [preloadedState] (any): 初始时的 state。 在同构应用中，你可以决定是否把服务端传来的 state 水合（hydrate）后传给它，或者从之前保存的用户会话中恢复一个传给它。如果你使用 combineReducers 创建 - reducer，它必须是一个普通对象，与传入的 keys 保持同样的结构。否则，你可以自由传入任何 reducer 可理解的内容。
    - enhancer (Function): Store enhancer 是一个组合 store creator 的高阶函数，返回一个新的强化过的 store creator。这与 middleware 相似，它也允许你通过复合函数改变 store 接口。
    - 返回值 (Store): 保存了应用所有 state 的对象。改变 state 的惟一方法是 dispatch action。你也可以 subscribe 监听 state 的变化，然后更新 UI。
- reducer

    - reducer(state,action)

- Store
    - getState()
    - dispatch(action)
    - subscribe(listener)
    - replaceReducer(nextReducer)

- combineReducers(reducers) 将 reducer 函数拆分成多个单独的函数，拆分后的每个函数负责独立管理 state 的一部分

- applyMiddleware(...middlewares) 中间件

react-redux

-
- connect();

异步操作中间件

- redux-thunk

练习

总结

下节课内容

- antd 使用
- CNode 项目实战