博学谷
www.boxuegu.com

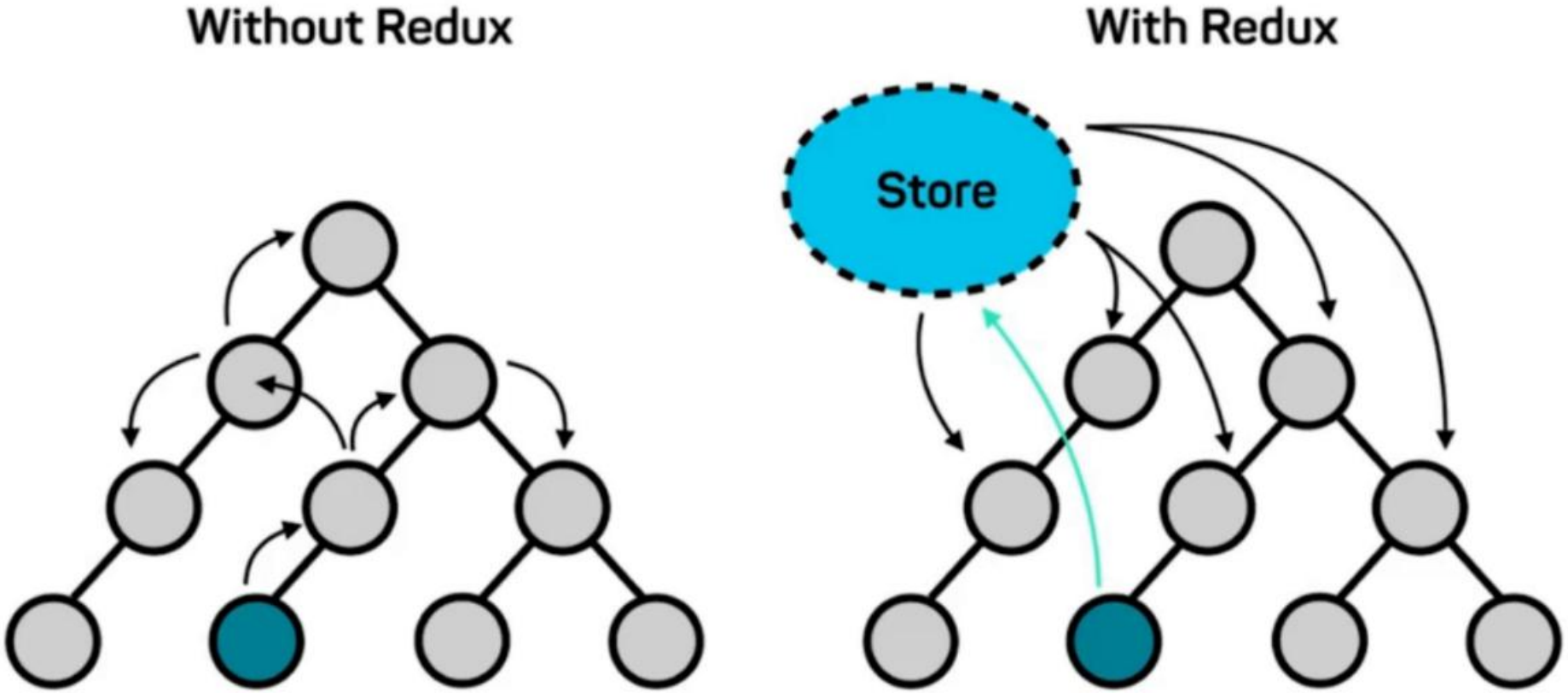
第四章 Redux应用与原理剖析

一样的在线教育，不一样的教学品质

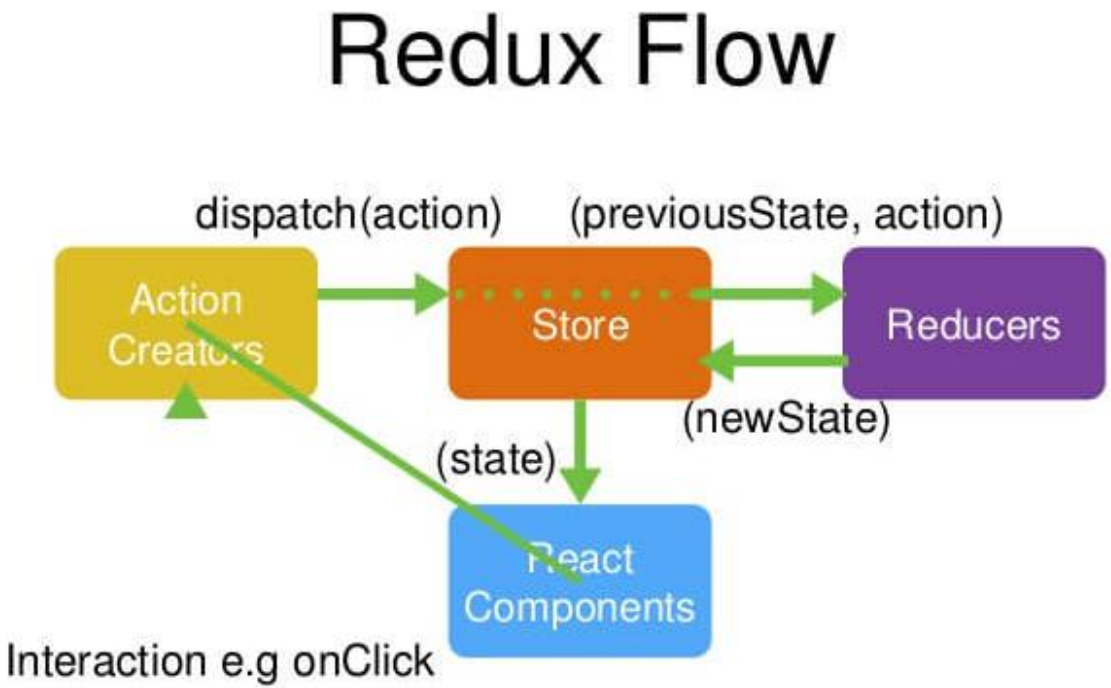
博学谷
www.boxuegu.com

目录 Contents

1. Redux简介



1. Redux简介


Redux Flow
dispatch(action)
(previousState, action)
Action Creators
Store
Reducers
(newState)
(state)
React Components
Interaction e.g onClick

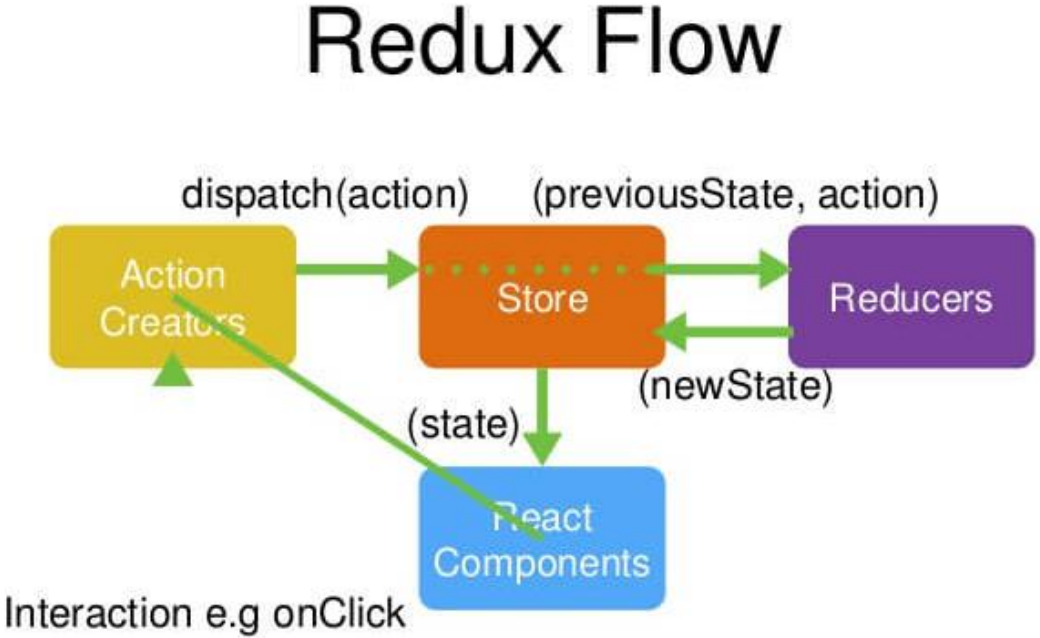
博学谷
www.boxuegu.com

目录 Contents

- Redux简介
- Redux基本使用
- Redux核心API理解
- 拆分reducer
- 中间件
- Redux 结合 React 使用
- Redux综合案例应用
- Redux源码解读

1. Redux基本使用

安装：npm install redux



博学谷
www.boxuegu.com

目录 Contents

1. Redux核心API理解

1.1 Store

Store 就是保存数据的地方，你可以把它看成一个容器。整个应用只能有一个 Store。

Redux 提供createStore这个函数，用来生成 Store。

```
import { createStore } from 'redux';

const store = createStore(fn);
```

1. Redux核心API理解

1.2 State

获取当前时刻的 State，可以通过store.getState( )拿到。

```
import { createStore } from 'redux';

const store = createStore(fn);

const state = store.getState();
```

1. Redux核心API理解

1.3 Action

Action 是一个对象。其中的type属性是必须的，表示 Action 的名称。其他属性可以自由设置。

```
const action = {
  type: 'ADD_TODO',
  payload: 'Learn Redux'
};
```

上面代码中，Action 的名称是ADD_TODO，它携带的信息是字符串Learn Redux。

可以这样理解，Action 描述当前发生的事情。改变 State 的唯一办法，就是使用 Action。它会运送数据到 Store。

1. Redux核心API理解

1.4 Action Creator

View 要发送多少种消息，就会有多少种 Action。可以定义一个函数来生成 Action，这个函数就叫 Action Creator。

```
const ADD_TODO = '添加 TODO';

function addTodo(text) {
  return {
    type: ADD_TODO,
    text
  }
}

const action = addTodo('Learn Redux');
```

1. Redux核心API理解

1.5 store.dispatch

store.dispatch()是 View 发出 Action 的唯一方法。

```
import { createStore } from 'redux';
const store = createStore(fn);

store.dispatch({
  type: 'ADD_TODO',
  payload: 'Learn Redux'
});
```

上面代码中，store.dispatch接受一个 Action 对象作为参数，将它发送出去。

1. Redux核心API理解

1.6 reducer

Store 收到 Action 以后，必须给出一个新的 State，这样 View 才会发生变化。这种 State 的计算过程就叫做 Reducer。

Reducer 是一个函数，它接受 Action 和当前 State 作为参数，返回一个新的 State。

```
const reducer = function (state, action) {
  // ...
  return new_state;
};
```

实际应用中，通过store.dispatch方法会触发 Reducer 的自动执行。为此，Store 需要知道 Reducer 函数，做法就是在生成 Store 的时候，将 Reducer 传入createStore方法。

```
import { createStore } from 'redux';
const store = createStore(reducer);
```

1. Redux核心API理解

1.7 store.subscribe()

Store 允许使用store.subscribe方法设置监听函数，一旦 State 发生变化，就自动执行这个函数。

```
import { createStore } from 'redux';

const store = createStore(reducer);

store.subscribe(listener);
```

显然，只要把 View 的更新函数（对于 React 项目，就是组件的render方法或setState方法）放入listen，就会实现 View 的自动渲染。

目录 Contents

1.拆分reducer

Reducer 函数负责生成 State。由于整个应用只有一个 State 对象，包含所有数据，对于大型应用来说，这个 State 必然十分庞大，导致 Reducer 函数也十分庞大。

```
const chatReducer = (state = defaultState, action = {}) => {
  const { type, payload } = action;
  switch (type) {
    case ADD_CHAT:
      return Object.assign({}, state, {
        chatLog: state.chatLog.concat(payload)
      });
    case CHANGE_STATUS:
      return Object.assign({}, state, {
        statusMessage: payload
      });
    case CHANGE_USERNAME:
      return Object.assign({}, state, {
        userName: payload
      });
    default: return state;
  }
};
```

1.拆分reducer

把 Reducer 函数拆分。不同的函数负责处理不同属性，最终把它们合并成一个大的 Reducer 即可。

```
const chatReducer = (state = defaultState, action = {}) => {
  return {
    chatLog: chatLog(state.chatLog, action),
    statusMessage: statusMessage(state.statusMessage, action),
    userName: userName(state.userName, action)
  }
};
```

1.拆分reducer

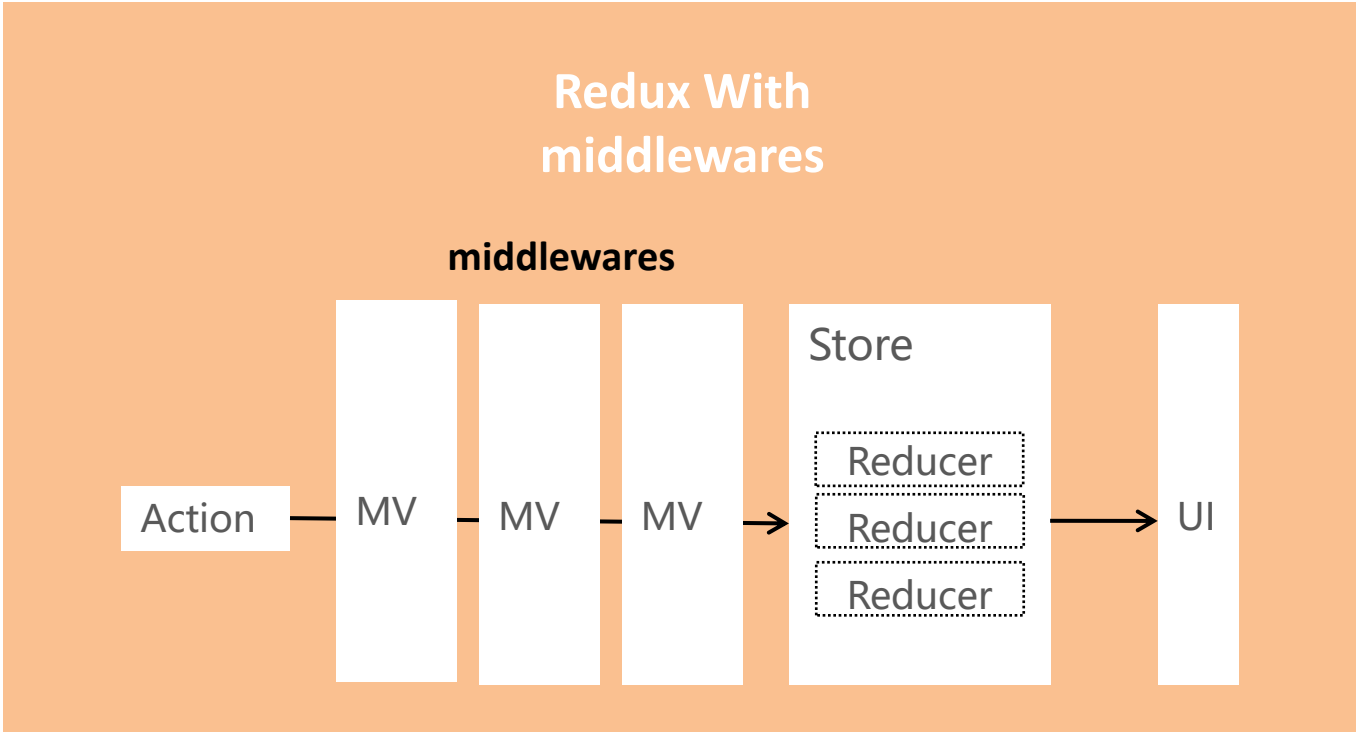
Redux 提供了一个combineReducers方法，用于 Reducer 的拆分。只要定义各个子 Reducer 函数，然后用这个方法，将它们合成一个大的 Reducer。

```
import { combineReducers } from 'redux';

const chatReducer = combineReducers({
  chatLog,
  statusMessage,
  userName
})

export default todoApp;
```

博学谷
www.boxuegu.com

目录 Contents

1.中间件



2. redux-logger中间件

redux-logger中间件是关于日志记录的中间件，用于开发的调试

地址： https://github.com/LogRocket/redux-logger

安装： npm i --save redux-logger

2. redux-thunk中间件

redux-thunk中间件可以实现异步的操作

redux-thunk安装如下:

```
npm install redux-thunk
```

地址：https://github.com/reduxjs/redux-thunk

目录 Contents

1. Redux 结合 React 使用

Redux 和 React 之间没有关系。

Redux 支持 React、Angular、Ember、jQuery 甚至纯 JavaScript。

Redux 默认并不包含 React 绑定库，需要单独安装。

```
npm install --save react-redux
```

Redux 的 React 绑定库基于 **容器组件**和**展示组件**相分离 的开发思想。

1. Redux 结合 React 使用

UI 组件有以下几个特征:

- 只负责 UI 的呈现，不带有任何业务逻辑
- 没有状态（即不使用this.state这个变量）
- 所有数据都由参数（this.props）提供
- 不使用任何 Redux 的 API

1. Redux 结合 React 使用

容器组件的特征:

- 负责管理数据和业务逻辑，不负责 UI 的呈现
- 带有内部状态
- 使用 Redux 的 API

总结：UI 组件负责 UI 的呈现，容器组件负责管理数据和逻辑。

目录 Contents

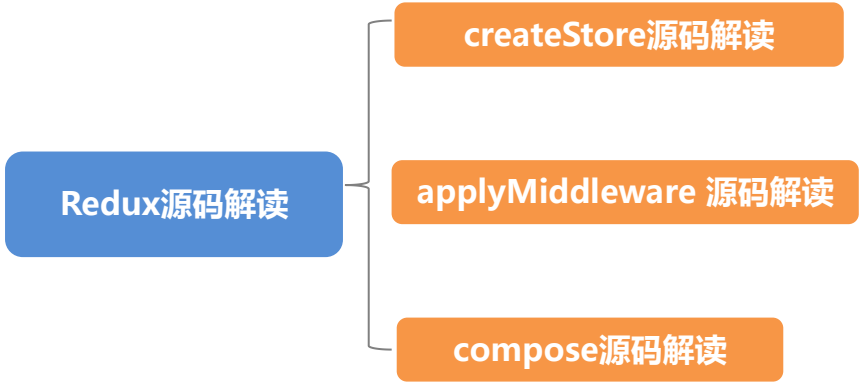
1. Redux综合案例应用

博学谷
www.boxuegu.com

目录 Contents

- Redux简介
- Redux基本使用
- Redux核心API理解
- 拆分reducer
- 中间件
- Redux 结合 React 使用
- Redux综合案例应用
- Redux源码解读

1. Redux源码解读


Redux源码解读
createStore源码解读
applyMiddleware 源码解读
compose源码解读

一样的在线教育，不一样的教学品质