如下 4 个步骤描述了redux的state是如何被更新的

****调用 [store.dispatch(action)](https://www.redux.org.cn/docs/api/Store.html" \l "dispatch)****

**调用store.dispatch(action)，将action传给store**

你可以在任何地方调用 [store.dispatch(action)](https://www.redux.org.cn/docs/api/Store.html" \l "dispatch)，

****Redux store 调用 reducer 函数。****

store会调用[reducer](https://www.redux.org.cn/docs/basics/Reducers.html)，并传入state 树和 action

根 reducer 可能接收到这样的数据：

// 当前应用的 state

let previousState = {

visibleTodoFilter: 'SHOW\_ALL',

todos: [

{

text: 'Read the docs.',

complete: false

}

]

}

// 将要执行的 action（添加一个 todo）

let action = {

type: 'ADD\_TODO',

text: 'Understand the flow.'

}

// reducer 返回处理后的 state

let nextState = todoApp(previousState, action)

****根 reducer 调用遍历所有 子reducer****

**各个reducer都会返回各自的state，组合成state树**

function todos(state = [], action) {

// 省略处理逻辑...

// 如果有更新，返回新的state

return nextState

}

function visibleTodoFilter(state = 'SHOW\_ALL', action) {

// 省略处理逻辑...

return nextState

}

let todoApp = combineReducers({

// 调用子reducer

todos,

visibleTodoFilter

})

****Redux store 保存了根 reducer 返回的完整 state 树****

这个新的state树就是应用的下一个 state！所有订阅 [store.subscribe(listener)](https://www.redux.org.cn/docs/api/Store.html" \l "subscribe) 的监听器都将被调用；监听器里可以调用 [store.getState()](https://www.redux.org.cn/docs/api/Store.html" \l "getState) 获得当前 state。