# https://segmentfault.com/a/1190000004236064?utm\_source=Weibo

# 1.创建action

# 2.创建 数据处理器

# combineReducers(reducers)

3创建 数据仓库（数据仓库工作链）

store 能维持应用的 state，并在当你发起 action 的时候调用 reducer。

* 维持应用的 state；
* 提供 [getState()](http://cn.redux.js.org/docs/api/Store.html" \l "getState) 方法获取 state；
* 提供 [dispatch(action)](http://cn.redux.js.org/docs/api/Store.html#dispatch) 方法更新 state；
* 通过 [subscribe(listener)](http://cn.redux.js.org/docs/api/Store.html#subscribe) 注册监听器;
* 通过 [subscribe(listener)](http://cn.redux.js.org/docs/api/Store.html#subscribe) 返回的函数注销监听器。

**createStore(reducer, [preloadedState],enhancer)**

创建一个 Redux [store](http://cn.redux.js.org/docs/api/Store.html) 来以存放应用中所有的 state。  
应用中应有且仅有一个 store。

参数：

reducer (Function): 接收两个参数，分别是当前的 state 树和要处理的 [action](http://cn.redux.js.org/docs/Glossary.html#action)，返回新的 [state 树](http://cn.redux.js.org/docs/Glossary.html#state)。

# applyMiddleware(...middlewares)

const store = createStore(

reducer,

preloadedState,

applyMiddleware(...middleware)

)

[Redux Thunk middleware](https://github.com/gaearon/redux-thunk)

reducer 中不能进行IO操作，所以需要直接执行 fetch操作。

## 陷阱1：

case DevPlanFormAction.GET\_DEV\_PLAN\_FORM\_REP\_ACTION:

\_state.dev\_plan = action.data;

const {dev\_plan} = \_state;

dev\_plan.tester = dev\_plan.tester\_owner.split(',');

dev\_plan.dev\_owner = dev\_plan.dev\_owner.split(',');

dev\_plan.watcher = dev\_plan.watcher?dev\_plan.watcher.split(','):[];

dev\_plan.related\_apps = dev\_plan.related\_apps.split(',');

return Object.assign({}, state, \_state);

const mapStateToProps = (state) => {

return {

depend\_list: state.agent[API.DEV\_PLAN.VERIFY\_CIRCLE\_DEPEND],

dev\_plan: state.dev\_plan\_form.dev\_plan,

}

};

**dev\_plan 其实没有改变，此时组件关心的是dev\_plan**

**redux又做state了优化，不会render**

陷阱2

不要操作 props，绝对不能改变props

生命周期中的preProps 可能失效

陷阱3

semantic-ui-react 的ref 被抛弃了

特别是this.refs.xx 在未来将可能被抛弃

减少使用ref，除非在特效、动画、moive等必须的情况下

原因

function 组件没有实例，返回的ref实例也是空的

破坏组件封装性

获取的是组件实例，不是DOM节点

**You may not use the ref attribute on functional components** because they don't have instances

如果想要用ref， 改成Component写法、或者在function中初始化一个指向内部组件的ref

你可以用下面方式处理

function CustomTextInput(props) {

return (

<div>

<input ref={props.inputRef} />

</div>

);

}

class Parent extends React.Component {

render() {

return (

<CustomTextInput

inputRef={el => this.inputElement = el}

/>

);

}

}

如果你没有组件控制权，想获取这个组件，只能用 [findDOMNode()](https://facebook.github.io/react/docs/react-dom.html" \l "finddomnode)

https://facebook.github.io/react/docs/refs-and-the-dom.html

https://github.com/gaearon/redux-devtools

https://github.com/gaearon/redux-devtools/blob/master/docs/Walkthrough.md

https://github.com/gaearon/redux-devtools#custom-monitors

http://www.ruanyifeng.com/blog/2016/09/redux\_tutorial\_part\_two\_async\_operations.html

http://mp.weixin.qq.com/s?\_\_biz=MzAwODcwODYwMw==&mid=2247483910&idx=1&sn=f5a0bbeef0abdfdc5b18d045ad0cec87&chksm=9b6b8a92ac1c0384004de77a1d4b801413829dc76304d37e9a47ff1b648f2ac0f0dfa45cb3ba&scene=0#rd

http://www.10tiao.com/html/481/201702/2653285438/1.html

http://2ality.com/2015/12/webpack-tree-shaking.html