React传值只有三种情况：父传子，子传父，兄弟之间传递。不管哪种情况传，最核心的都是要通过props传递

State分为三种：一种是组件内部的state，一种是同步（reducer）时UI组件之间传递的state，另一种是异步后台（effect）返回的state

Redux：redux就是一个js容器，用于管理全局组件的state状态。

redux三大核心：1单一数据源、2state是只读的、3使用纯函数来执行修改



1. 单一数据源：整个应用的数据源被存储在一颗object tree中，并且这个object tree只存在于唯一一个store中
2. 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。
3. 使用纯函数（一般起名Reducer）来执行修改（纯函数为了保证复用性）。为了描述action如何改变state tree，你需要去编写reducers。Reducer本质是一个函数，它用来响应发送过来的actions，然后经过处理，把state发送给store。注意：在reducer中需要return返回值，这样store才能接收到数据。并且reducer函数会接受到两个参数，第一个参数是初始化state，第二个是action。
4. Action事件。Action是把数据从应用传到store的载体，它是store数据的唯一来源。一般通过store.dispatch()将action传递给store。Action的本质是js的一个普通对象。Action对象内部必须要有一个type属性来表示要执行的动作
5. Store就是把action与reducer联系到一起的对象。Store作用：1维持应用的state。2提供getstate()方法来获取到state。3提供dispatch()方法发送action。4通过subscribe()来注册监听。5通过subscribe()返回值类注销监听

React-redux：最大特点是解决每次都要创建一个store的问题。两个重要概念：1provider，这是一个组件，能够使你整个app都能获取到store数据；2connect()，这个方法能够使组件跟store来进行关联



1. provider包裹在根组件最外层，使所有的子组件都可以拿到store。
2. Provider实现原理：provider接受store作为props，然后通过context往下传递，这样react中任何组件都可以用过context获取到store
3. React-redux中也要引入redux插件，因为react-redux中的store和createStore要从redux中生成
4. 引入provider组件：import {Provider} from ‘react-redux’；然后利用provider组件将我们整个结果进行包裹并且传递引入的store属性。如

Function App(){

Return (<Provider store={store}>......</Provider >)

}

1. 引入connect方法：import connect from ‘react-redux’；
2. 调用connect方法：connect（...）（组件名）
3. Connect的两个最重要参数：1mapStateToProps（state，ownProps）。2mapDispatchToProps(dispatch,ownProps)。
4. mapStateToProps（state，ownProps）（接收）。该函数何时使用：当你的组件想要接收store中的数据时，就调用该参数。注意这个参数必须有返回值，也是通过props将store的值传递到组件。其中state为redux中的store，ownProps为自己的props
5. mapDispatchToProps(dispatch,ownProps)（发送）。该函数何时使用：如果你哪个组件想要发送action，那么就调用该方法。将action作为props绑定到我们自己的函数中。其中：dispatch===store.dispatch()，ownProps为自己的props

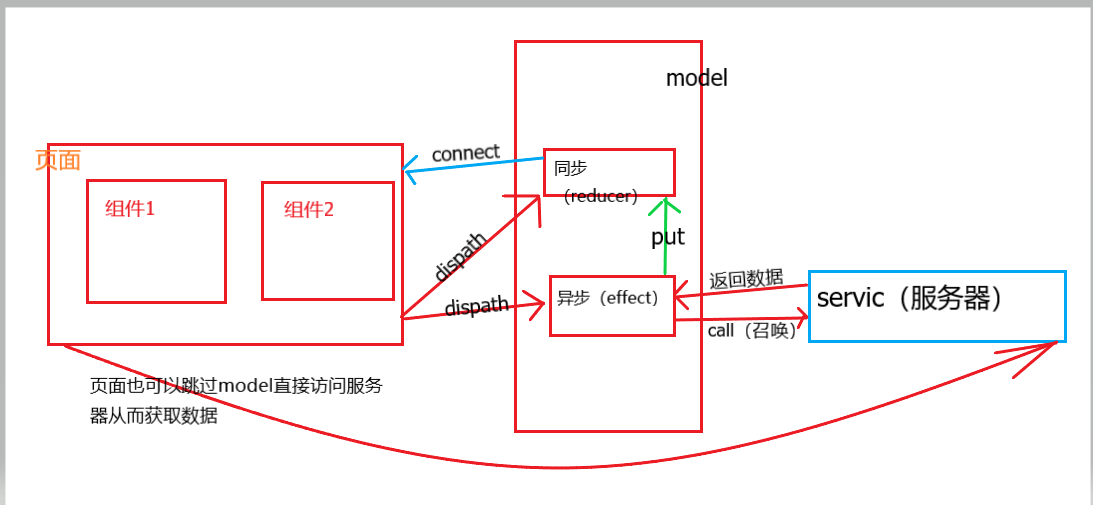
ComA发送action：1、导入connect。2、利用connect对组件进行加强：connect（要接收数据的函数（一般设置成null），要发送action的函数）（要加强的组件（ComA））。3、实现connect第二个参数：构建一个函数mapDispatchToProps（dispatch），dispatch就是用来发送action的。4、mapDispatchToProps这个函数里要返回一个对象，key是方法名、value是调用dispatch去发送action。5组件的内部，就可以通过this.props来拿到这个方法

ComB接收state：1、导入connect。2、利用connect对组件进行加强：connect（要接收state的函数（mapStateToProps））（要加强的组件（ComB））。3、实现mapStateToProps，注意其中一个参数为state，把这个state return。组件内部通过this.props拿到这个数据

1. 用react-redux构建全局状态管理的基本步骤
2. 创建reducer，在函数里面判断action的type属性
3. 利用createStore构建store（要用到redux插件）
4. 在App.js中导入store
5. 在App.js中导入Provider（用到react-reudx插件），记得设置store属性
6. 在要接收和要发送的子组件中，分别导入connect并实现connect方法。注意发送方的connect（null,...）（组件名）的第一个括号中的第一个参数设置为null
7. 实现connect中的参数函数

Redux-saga：常见API:createSagaMiddleware(options)、middleware.run(saga,...args)

DvaJs：redux+redux-saga（既简化redux也能异步操作）。Dva实际上也是要通过props来传递数据的（）



数据分为2种数据：1是公共数据（这个数据多个组件共用），2是私有数据（这个数据只在这个组件里使用）。Redux中把公共数据叫做store（也叫state），dva中把公共数据叫做model。Model里的数据又分为两种：同步的数据（reducer）、异步的数据（effect）

1. 如果组件（页面）发起的是异步请求，异步回调的数据是不能直接返回给组件（页面）的，要先调给同步，再把数据从同步返回到组件（页面）中
2. 异步（effect）干嘛的？调接口！就是那个一直不懂的概念！！！调用异步（effect），然后effect调（访问）服务器（service），然后服务（service）返回数据给effect，然后再put给reducer，reducer再connect回页面
3. dva中的action存放于dispatch里