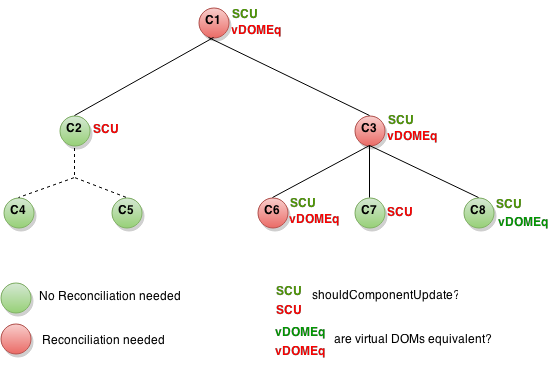
**1. React 的渲染机制————Reconciliation**

如果某个props或者state发生了变化，引起相应元素发生变化，则该元素以及该元素的所有父元素的shouldComponentUpdate 都会返回true，都会被标记为需要进行diff比对，并且父而子一个个寻找真正需要重新渲染的元素，确认不是所需寻找的对象以后直接跳过，如果是需要重新渲染的元素，则删除原DOM树该元素以及该元素的所有子节点，并复制新DOM树上该元素以及该元素的所有子节点，重新渲染。



React 为函数组件提供了一个 memo 方法，它和 PureComponent 在数据比对上唯一的区别就在于 只进行了 props 的浅比较，因为函数组件是没有 state 的。而且它的用法很简单，直接将函数传入 memo 中导出即可。形如:

**function** **Home** () {

//xxx

} **export** **default** React.memo (Home);

**2.路由**

通过一个路由的配置文件设定好路由结构，在该路由文件中引用所有节点所需要用到的组件，最后在根节点通过声明

{ renderRoutes(routes) }

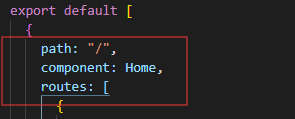
使用路由。

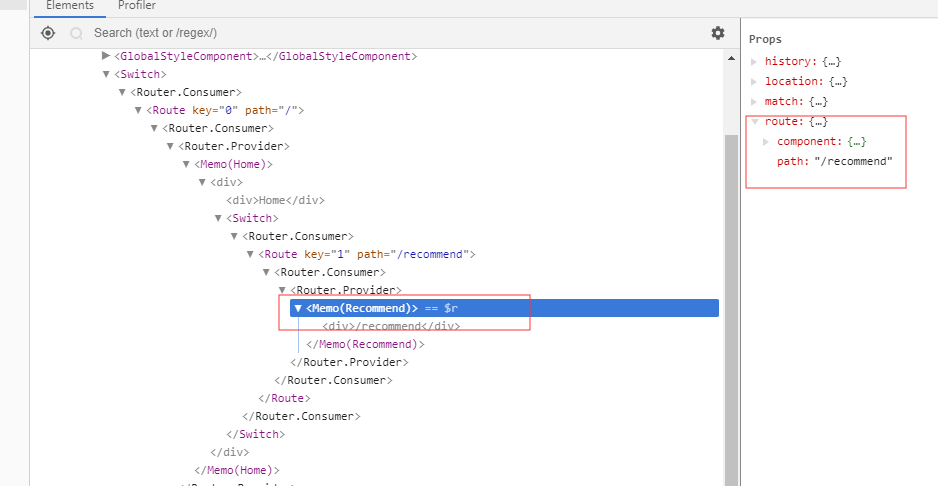
重点：嵌套路由。需要在产生嵌套的父组件内再次声明

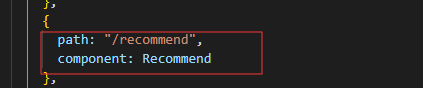
{ renderRoutes(routes) }

这里有个重点，每个组件的props默认都携带了route对象，且route对象的内容与react-route-config文件中的内容保持一致。









1. 图标的使用

<https://segmentfault.com/a/1190000016235900>

1. Redux

使用immutable的目的在于性能优化，只需要momo进行浅层比较即可完成diff过程。