Redux是流行框架React不折不扣的最佳伴侣。React管理显示，贤内助Redux管理数据。

几个主要概念

State

State直译为状态。它在Redux里就是数据模型，或者说是当前的某一部分数据。

这些数据是用于应用的显示，所以可以说这些数据就代表了应用的状态。

比如最简单的Todo list，每一个todo包含了todo内容、状态（完成、未完成）。这里每个todo可以称为todo state，而这个包含todo顺序的列表又可以称为todo list state。

Store

Store就是整个应用的数据仓库，它是State的集合，也是连接Action、Reducer、State这几个主要构成的桥梁。

Action

这里可以看出Action其实是store.dispatch 方法传递的指令和相关参数、数据。

也就是说store.dispatch 会将Action指令发送给所有挂载在Store上的Reducer，让Reducer根据指令和“Action”提供的相关数据处理更新现有的State。

简单来说，Action只是用来告诉Reducer发生了什么，这些是你可能需要用的数据。然后它就啥都不管了。

由于Redux作者给出的关于Action的实例通常如下：

store.dispatch({

type: TODO.COMPLETE\_TODO

data: 'xxx'

});

Reducer

如Action一节中所说，Reducer负责接收store.dispatch(action)发送过来的指令和相关数据，并且对现有State进行处理，并返回处理过后的State，更新我们的Store。

注意上一节中我将Reducer比做发布/订阅模式里的订阅者。但它们并不相同。

不论dispatch了什么Action，所有挂载在Store上的Reducer都能依次接收到，并且可以对State进行处理，全看你代码怎么写。自由度很大。

Selector

Selector是用来从Store中选出显示或者做其他处理通常所需要的一些数据，并做一定的整理。

Redux其实就是包装了一个"客户端数据库"出来。

Store => 数据库

State => 数据表

Action => 查询指令

Reducer => 存储过程、数据锁等

Selector => 数据库视图

通过Redux进行组件的流畅通信：

