React.js 小书

<-- 返回首页

动手实现 React-redux（五）：Provider

- 作者：胡子大哈
- 原文链接： http://huziketang.com/books/react/lesson40

（本文未审核）

我们要把 context 相关的代码从所有业务组件中清除出去，现在的代码里面还有一个地方是被污染的。那就是 `src/index.js` 里面的 `Index`：

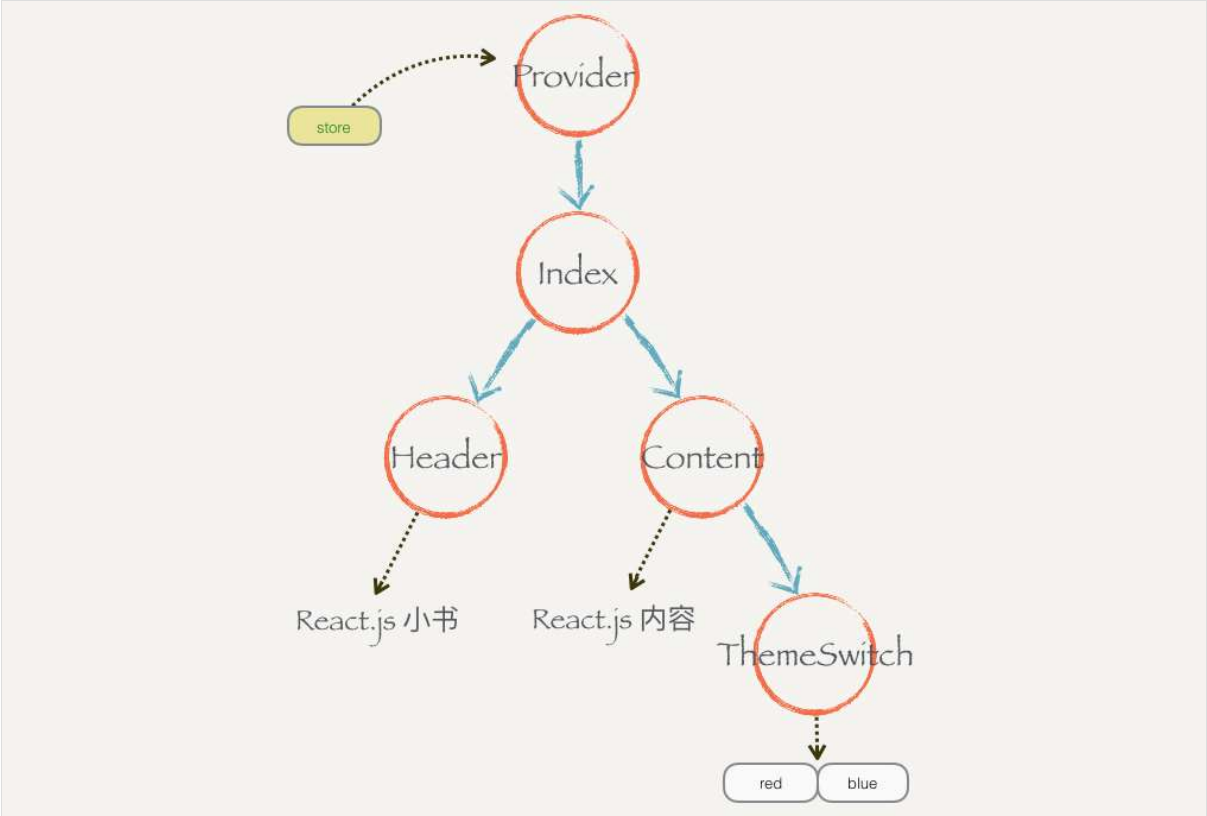
```
...
class Index extends Component {
  static childContextTypes = {
    store: PropTypes.object
  }

  getChildContext () {
    return { store }
  }

  render () {
    return (
      <div>
        <Header />
        <Content />
      </div>
    )
  }
}
...
```

其实它要用 context 就是因为要把 `store` 存放到里面，好让子组件 `connect` 的时候能够取到 `store`。我们可以额外构建一个组件来做这种脏活，然后让这个组件成为组件树的根节点，那么它的子组件都可以获取到 context 了。

我们把这个组件叫 `Provider`，因为它提供（provide）了 `store`：



在 `src/react-redux.js` 新增代码：

```
export class Provider extends Component {
  static propTypes = {
    store: PropTypes.object,
    children: PropTypes.any
  }

  static childContextTypes = {
    store: PropTypes.object
  }

  getChildContext () {
    return {
      store: this.props.store
    }
  }

  render () {
    return (
      <div>{this.props.children}</div>
    )
  }
}
```

`Provider` 做的事情也很简单，它就是一个容器组件，会把嵌套的内容原封不动作为自己的子组件渲染出来。它还会把外界传给它的 `props.store` 放到 context，这样子组件 `connect` 的时候都可以获取到。

可以用它来重构我们的 `src/index.js`：

```
...
// 头部引入 Provider
import { Provider } from './react-redux'
...

// 删除 Index 里面所有关于 context 的代码
class Index extends Component {
  render () {
    return (
      <div>
        <Header />
        <Content />
      </div>
    )
  }
}

// 把 Provider 作为组件树的根节点
ReactDOM.render(
  <Provider store={store}>
    <Index />
  </Provider>,
  document.getElementById('root')
)
```

这样我们就把所有关于 context 的代码从组件里面删除了。

> 因为第三方评论工具有问题，对本章节有任何疑问的朋友可以移步到 React.js 小书的论坛 发帖，我会回答大家的疑问。

下一节：动手实现 React-redux（六）：React-redux 总结

上一节：动手实现 React-redux（四）：mapDispatchToProps