一. React的原理

在Web开发中，我们需要将变化的数据实时反应到UI上，这时就需要对DOM进行操作。而复杂或频繁的DOM操作通常是性能瓶颈产生的原因（如何进行高性能的复杂DOM操作通常是衡量一个前端开发人员技能的重要指标）。React为此引入了虚拟DOM（Virtual DOM）的机制：在浏览器端用Javascript实现了一套DOM API。基于React进行开发时所有的DOM构造都是通过虚拟DOM进行，每当数据变化时，React都会重新构建整个DOM树，然后React将当前整个DOM树和上一次的DOM树进行对比，得到DOM结构的区别，然后仅仅将需要变化的部分进行实际的浏览器DOM更新。而且React能够批处理虚拟DOM的刷新，在一个事件循环（Event Loop）内的两次数据变化会被合并，例如你连续的先将节点内容从A变成B，然后又从B变成A，React会认为UI不发生任何变化，而如果通过手动控制，这种逻辑通常是极其复杂的。尽管每一次都需要构造完整的虚拟DOM树，但是因为虚拟DOM是内存数据，性能是极高的，而对实际DOM进行操作的仅仅是Diff部分，因而能达到提高性能的目的。这样，在保证性能的同时，开发者将不再需要关注某个数据的变化如何更新到一个或多个具体的DOM元素，而只需要关心在任意一个数据状态下，整个界面是如何Render的。

二. React的的优缺点：

好处：1.组件化——分工、合作

2.虚拟DOM——性能高

3.跨平台——移动端

缺点：1.学习曲线特别陡

2.设计思想特别

React技术栈

1.React主体

2.WebPack：grunt、gulp自动化构建工具

3.Flex：布局

4.React-Route：路由

5.Redux：View层

6.Mocha：测试

7.Istanbul：覆盖率

8.JSX

下面是来源网上的一张图



三、JSX介绍

JSX是React的核心组成部分，它使用XML标记的方式去直接声明界面，界面组件之间可以互相嵌套。可以理解为在JS中编写与XML类似的语言,一种定义带属性树结构（DOM结构）的语法，它的目的不是要在浏览器或者引擎中实现，它的目的是通过各种编译器将这些标记编译成标准的JS语言。

babel：JSX解析器，把JSX转化成JS语法，这个工作会由babel自动完成。同时引入babel后，你就可以使用新的es6语法，babel会帮你把es6语法转化成es5语法，兼容更多的浏览器。

四、组件化的开发思路

对于React从功能的角度出发，将UI分成不同的组件，每个组件都独立封装，应该具有如下特征：

1、可组合（Composeable）：一个组件易于和其它组件一起使用，或者嵌套在另一个组件内部。如果一个组件内部创建了另一个组件，那么说父组件拥有（own）它创建的子组件，通过这个特性，一个复杂的UI可以拆分成多个简单的UI组件；

2、可重用（Reusable）：每个组件都是具有独立功能的，它可以被使用在多个UI场景；

3、可维护（Maintainable）：每个小的组件仅仅包含自身的逻辑，更容易被理解和维护；

4、可测试（Testable）：因为每个组件都是独立的，那么对于各个组件分别测试显然要比对于整个UI进行测试容易的多。

五、组件-生存周期

componentWillMount() 创建之前

componentDidMount() 创建之后

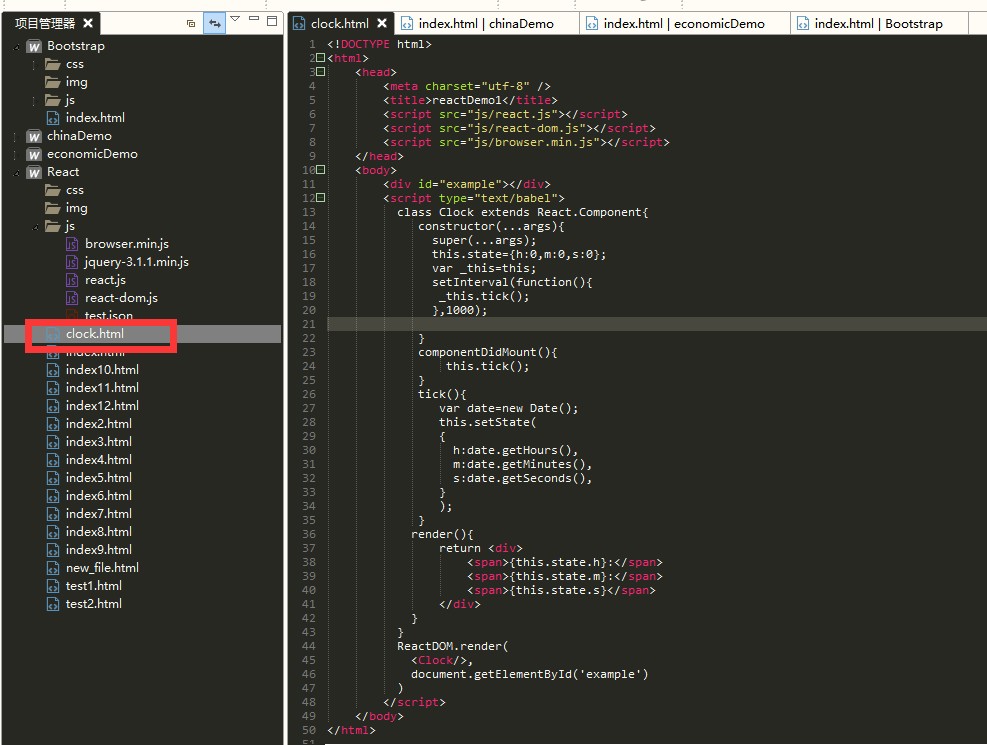
componentWillupdate() 更新之前

componentDidupdate() 更新之后

componentWillUnmount() 卸载之前

componentWillReceiveProps() 组件参数更新

六、React组件开发的例子-简单秒表



七、结论

React是一个全新思路的前端UI框架，它完全接管了UI开发中最为复杂的局部更新部分，擅长在在复杂场景下保证高性能；同时，它引入了基于组件的开发思想，从另一个角度来重新审视UI的构成。通过这种方法，不仅能够提高开发效率，而且可以让代码更容易理解，维护和测试