现在最热门的前端框架有AngularJS、React、Bootstrap等。ReactJs的虚拟DOM（Virtual DOM）和组件化的开发，下面来跟我一起领略

****一、ReactJS简介****

React 起源于 Facebook 的内部项目，因为该公司对市场上所有 JavaScript MVC 框架，都不满意，就决定自己写一套，用来架设 Instagram 的网站。做出来以后，发现这套东西很好用，就在2013年5月开源了。由于 React 的设计思想极其独特，属于革命性创新，性能出众，代码逻辑却非常简单。所以，越来越多的人开始关注和使用，认为它可能是将来 Web 开发的主流工具。

ReactJS官网地址：[http://facebook.github.io/react/](http://facebook.github.io/react/" \t "http://www.cocoachina.com/webapp/20150721/_blank)

Github地址：[https://github.com/facebook/react](https://github.com/facebook/react" \t "http://www.cocoachina.com/webapp/20150721/_blank)

****二、对ReactJS的认识及ReactJS的优点****

首先，对于React，有一些认识误区，这里先总结一下：

React不是一个完整的MVC框架，最多可以认为是MVC中的V（View），甚至React并不非常认可MVC开发模式；

React的服务器端Render能力只能算是一个锦上添花的功能，并不是其核心出发点，事实上React官方站点几乎没有提及其在服务器端的应用；

有人拿React和Web Component相提并论，但两者并不是完全的竞争关系，你完全可以用React去开发一个真正的Web Component；

React不是一个新的模板语言，JSX只是一个表象，没有JSX的React也能工作。

****1、ReactJS的背景和原理****

在Web开发中，我们总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（ 命令比较文本文件。它能比较单个文件或者目录内容。diff 命令只有当输入为文本文件时才有效）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。

如果你像在90年代那样写过服务器端Render的纯Web页面那么应该知道，服务器端所要做的就是根据数据Render出HTML送到浏览器端。如果这时因为用户的一个点击需要改变某个状态文字，那么也是通过刷新整个页面来完成的。服务器端并不需要知道是哪一小段HTML发生了变化，而只需要根据数据刷新整个页面。换句话说，任何UI的变化都是通过整体刷新来完成的。而React将这种开发模式以高性能的方式带到了前端，每做一点界面的更新，你都可以认为刷新了整个页面。至于如何进行局部更新以保证性能，则是React框架要完成的事情。

基于React的开发思路，你永远只需要关心数据整体，两次数据之间的UI如何变化，则完全交给框架去做。可以看到，使用React大大降低了逻辑复杂性，意味着开发难度降低，可能产生Bug的机会也更少。

****2、组件化****

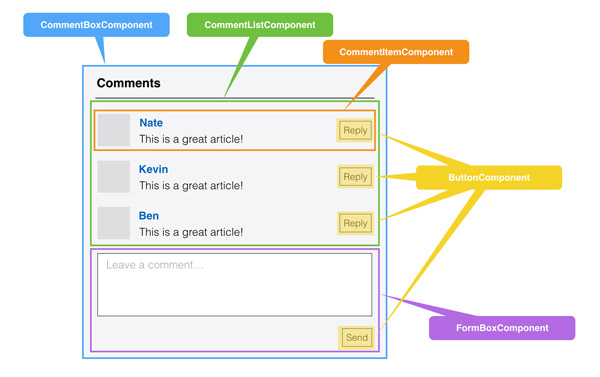
虚拟DOM(virtual-dom)不仅带来了简单的UI开发逻辑，同时也带来了组件化开发的思想，所谓组件，即封装起来的具有独立功能的UI部件。React推荐以组件的方式去重新思考UI构成，将UI上每一个功能相对独立的模块定义成组件，然后将小的组件通过组合或者嵌套的方式构成大的组件，最终完成整体UI的构建。例如，Facebook的instagram.com整站都采用了React来开发，整个页面就是一个大的组件，其中包含了嵌套的大量其它组件，大家有兴趣可以看下它背后的代码。

如果说MVC的思想让你做到视图-数据-控制器的分离，那么组件化的思考方式则是带来了UI功能模块之间的分离。我们通过一个典型的Blog评论界面来看MVC和组件化开发思路的区别。

对于MVC开发模式来说，开发者将三者定义成不同的类，实现了表现，数据，控制的分离。开发者更多的是从技术的角度来对UI进行拆分，实现松耦合。

对于React而言，则完全是一个新的思路，开发者从功能的角度出发，将UI分成不同的组件，每个组件都独立封装。

在React中，你按照界面模块自然划分的方式来组织和编写你的代码，对于评论界面而言，整个UI是一个通过小组件构成的大组件，每个组件只关心自己部分的逻辑，彼此独立。



React认为一个组件应该具有如下特征：

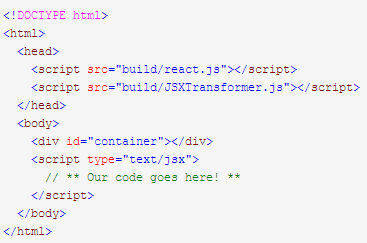
****（1）可组合（Composeable）****：一个组件易于和其它组件一起使用，或者嵌套在另一个组件内部。如果一个组件内部创建了另一个组件，那么说父组件拥有（own）它创建的子组件，通过这个特性，一个复杂的UI可以拆分成多个简单的UI组件；

****（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；

****三、下载ReactJS，编写Hello，world****

ReactJs下载非常简单，给出下载地址（[链接](http://facebook.github.io/react/downloads.html" \t "http://www.cocoachina.com/webapp/20150721/_blank)：http://facebook.github.io/react/downloads.html）,下载完成后，我么看到的是一个压缩包。解压后，我们新建一个html文件，引用react.js和JSXTransformer.js这两个js文件。html模板如下(js路径改成自己的):

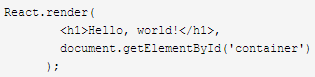


这里大家可能会奇怪，****为什么script的type是text/jsx,这是因为 React 独有的 JSX 语法，跟 JavaScript 不兼容。****凡是使用 JSX 的地方，都要加上 type="text/jsx" 。 其次，React 提供两个库： react.js 和 JSXTransformer.js ，它们必须首先加载。其中，JSXTransformer.js 的作用是将 JSX 语法转为 JavaScript 语法。这一步很消耗时间，实际上线的时候，应该将它放到服务器完成。

到这里我们就可以开始编写代码了，首先我们先来认识一下ReactJs里面的React.render方法：

React.render 是 React 的最基本方法，用于将模板转为 HTML 语言，并插入指定的 DOM 节点。

下面我们在script标签里面编写代码，来输出Hello，world，代码如下：



****这里需要注意的是，react并不依赖jQuery，当然我们可以使用jQuery，但是render里面第二个参数必须使用JavaScript原生的getElementByID方法，不能使用jQuery来选取DOM节点。****

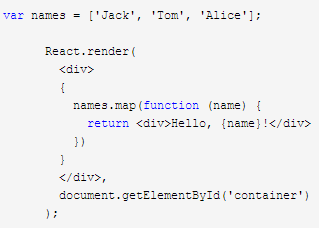
然后，在浏览器打开这个页面，就可以看到浏览器显示一个大大的Hello，world,因为我们用了

# 标签。

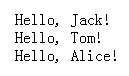
到这里，恭喜，你已经步入了ReactJS的大门~~下面，让我们来进一步学习ReactJs吧~~

****四、Jsx语法****

HTML 语言直接写在 JavaScript 语言之中，不加任何引号，这就是 JSX 的语法，它允许 HTML 与 JavaScript 的混写，了解过AngularJs的看到下面的代码一定会感觉很熟悉的，我们来看代码：



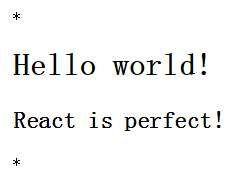
这里我们声明了一个names数组，然后遍历在前面加上Hello,输出到DOM中，输出结果如下：



JSX 允许直接在模板插入 JavaScript 变量。如果这个变量是一个数组，则会展开这个数组的所有成员，代码如下：



显示结果如下：



这里的星号只是做标识用的，大家不要被迷惑了~~

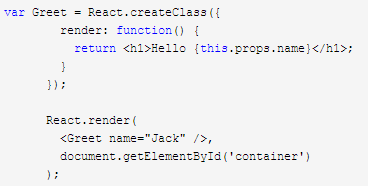
你看到这里，说明你对React还是蛮感兴趣的，恭喜你，坚持下来了，那么下面，我们开始学习React里面的"真功夫"了~~ Are　you ready?

****五、ReactJS组件****

****1、组件属性****

前面说了，ReactJS是基于组件化的开发，下面我们开始来学习ReactJS里面的组件，React 允许将代码封装成组件（component），然后像插入普通 HTML 标签一样，在网页中插入这个组件。React.createClass 方法就用于生成一个组件类。

下面，我们来编写第一个组件Greet，有一个name属性，然后输出hello + name的值，代码如下：



看到这段代码，接触过AngularJS的朋友们是不是有一种熟悉的感觉，不过这里有几点需要注意：

1）获取属性的值用的是this.props.属性名

2）创建的组件名称首字母必须大写。

3）为元素添加css的class时，要用className。

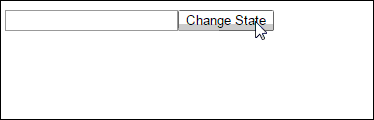
4）组件的style属性的设置方式也值得注意，要写成style={{width: this.state.witdh}}。

****2、组件状态****

组件免不了要与用户互动，React 的一大创新，就是将组件看成是一个状态机，一开始有一个初始状态，然后用户互动，导致状态变化，从而触发重新渲染 UI 。下面我们来编写一个小例子，一个文本框和一个button，通过点击button可以改变文本框的编辑状态，禁止编辑和允许编辑。通过这个例子来理解ReactJS的状态机制。先看代码：



这里，我们又使用到了一个方法getInitialState,这个函数在组件初始化的时候执行，必需返回NULL或者一个对象。这里我们可以通过this.state.属性名来访问属性值，这里我们将enable这个值跟input的disabled绑定，当要修改这个属性值时，要使用setState方法。我们声明handleClick方法，来绑定到button上面，实现改变state.enable的值。效果如下：



原理分析：

当用户点击组件，导致状态变化，this.setState 方法就修改状态值，每次修改以后，自动调用 this.render 方法，再次渲染组件。

这里值得注意的几点如下：

1）getInitialState函数必须有返回值，可以是NULL或者一个对象。

2）访问state的方法是this.state.属性名。

3）变量用{}包裹，不需要再加双引号。

****3、组件的生命周期****

组件的生命周期分成三个状态：

Mounting：已插入真实 DOM

Updating：正在被重新渲染

Unmounting：已移出真实 DOM

React 为每个状态都提供了两种处理函数，will 函数在进入状态之前调用，did 函数在进入状态之后调用，三种状态共计五种处理函数。

componentWillMount()

componentDidMount()

componentWillUpdate(object nextProps, object nextState)

componentDidUpdate(object prevProps, object prevState)

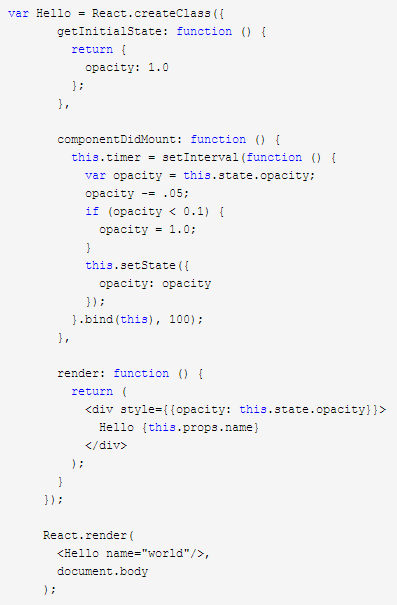
componentWillUnmount()

此外，React 还提供两种特殊状态的处理函数。

componentWillReceiveProps(object nextProps)：已加载组件收到新的参数时调用

shouldComponentUpdate(object nextProps, object nextState)：组件判断是否重新渲染时调用

下面来看一个例子：



上面代码在hello组件加载以后，通过 componentDidMount 方法设置一个定时器，每隔100毫秒，就重新设置组件的透明度，从而引发重新渲染。

****4、组件的嵌套****

React是基于组件化的开发，那么组件化开发最大的优点是什么？毫无疑问，当然是复用，下面我们来看看React中到底是如何实现组件的复用的，这里我们还写一个例子来说吧，代码如下：



这里我们创建了一个Search组件，然后又创建了一个Page组件，然后我们在Page组件中调用Search组件，并且调用了两次，这里我们通过属性searchType传入值，最终显示结果如图：

****六、ReactJS小结****

关于ReactJS今天就先学习到这里了，下面来总结一下，主要有以下几点：

1、ReactJs是基于组件化的开发，所以最终你的页面应该是由若干个小组件组成的大组件。

2、可以通过属性，将值传递到组件内部，同理也可以通过属性将内部的结果传递到父级组件(留给大家研究)；要对某些值的变化做DOM操作的，要把这些值放到state中。

3、为组件添加外部css样式时，类名应该写成className而不是class;添加内部样式时，应该是style={{opacity: this.state.opacity}}而不是style="opacity:{this.state.opacity};"。

4、组件名称首字母必须大写。

5、变量名用{}包裹，且不能加双引号。

****七、参考资料****

[React中文文档](http://reactjs.cn/" \t "http://www.cocoachina.com/webapp/20150721/_blank)

[React入门实例教程](http://www.ruanyifeng.com/blog/2015/03/react.html" \t "http://www.cocoachina.com/webapp/20150721/_blank)

[颠覆式前端UI开发框架：React](http://www.infoq.com/cn/articles/subversion-front-end-ui-development-framework-react" \t "http://www.cocoachina.com/webapp/20150721/_blank)