## 与 DOM 节点操作类插件结合

对于 React 之外的 DOM 节点操作，React 是不会去处理的，因为 React 内部有自己的渲染逻辑。当相同的 DOM 节点被外部的代码改变时，React 就会很迷茫，并不知道发生了什么。

但这也不意味着 React 无法与其他操作 DOM 节点的库一起使用，你只要清楚他们分别在做什么就可以了。

最简单的方式就是阻止 React 更新外部在操作的节点，那么你可以通过生成一个 React 根本不会去更新的元素来实现，比如空的 <div />。

### 进一步的解释

为了解释得更清楚，我们先来封装一个通用的 jQuery 插件吧。

在这里，我们给 DOM 的根节点元素加了一个 [ref](http://www.css88.com/react/docs/refs-and-the-dom.html)。在 componentDidMount 中，我们会调用这个 ref，并把它传给 jQuery 插件。

为了防止 React 在 DOM 加载后修改节点，我们先要在 render() 中返回一个空的 <div />。这个空的 <div /> 既没有属性也没有子元素，这样一来，React 就不会更新它了。那么，我们封装的 jQuery 插件就可以随意地更新这个节点。

class SomePlugin extends React.Component {

componentDidMount() {

this.$el = $(this.el);

this.$el.somePlugin();

}

componentWillUnmount() {

this.$el.somePlugin('destroy');

}

render() {

return <div ref={el => this.el = el} />;

}

}

为了防止内存泄漏，请务必在生命周期函数中移除插件挂载的事件监听器。

## 在 React 中使用其他引擎加载页面

### 在 React 中使用“字符串替换”类的库

早起的网页应用中，一个常用的方式是先把内容定义成字符串片段，然后插入到 DOM 节点中

$('#container').html('<button id="btn">Say Hello</button>');

$('#btn').click(function() {

alert('Hello!');

});

可以改写成这样的 React 组件：

function Button() {

return <button id="btn">Say Hello</button>;

}

ReactDOM.render(

<Button />,

document.getElementById('container'),

function() {

$('#btn').click(function() {

alert('Hello!');

});

}

);