组件生命周期

在开发应用时，理解路由组件的生命周期是非常重要的。 后面我们会以获取数据这个最常见的场景为例，介绍一下路由改变时，路由组件生命周期的变化情况。

路由组件的生命周期和 React 组件相比并没有什么不同。 所以让我们先忽略路由部分，只考虑在不同 URL 下，这些组件是如何被渲染的。

路由配置如下：

```
<Route path="/" component={App}>
  <IndexRoute component={Home}/>
  <Route path="invoices/:invoiceId" component={Invoice}/>
  <Route path="accounts/:accountId" component={Account}/>
</Route>
```

路由切换时，组件生命周期的变化情况

1. 当用户打开应用的 '/' 页面

组件	生命周期
App	`componentDidMount`
Home	`componentDidMount`
Invoice	N/A
Account	N/A

2. 当用户从 '/' 跳转到 '/invoice/123'

组件	生命周期
App	`componentWillReceiveProps`, `componentDidUpdate`
Home	`componentWillUnmount`
Invoice	`componentDidMount`
Account	N/A

- `App` 从 router 中接收到新的 props（例如 `children`、`params`、`location` 等数据), 所以 `App` 触发了 `componentWillReceiveProps` 和 `componentDidUpdate` 两个生命周期方法
- `Home` 不再被渲染，所以它将被移除
- `Invoice` 首次被挂载

3. 当用户从 `/invoice/123` 跳转到 `/invoice/789`

组件	生命周期
App	componentWillReceiveProps, componentDidUpdate
Home	N/A
Invoice	componentWillReceiveProps, componentDidUpdate
Account	N/A

所有的组件之前都已经被挂载， 所以只是从 router 更新了 props.

4. 当从 `/invoice/789` 跳转到 `/accounts/123`

组件	生命周期
App	componentWillReceiveProps, componentDidUpdate
Home	N/A
Invoice	componentWillUnmount
Account	componentDidMount

获取数据

虽然还有其他通过 router 获取数据的方法， 但是最简单的方法是通过组件生命周期 Hook 来实现。 前面我们已经理解了当路由改变时组件生命周期的变化， 我们可以在 `Invoice` 组件里实现一个简单的数据获取功能。

```
let Invoice = React.createClass({

  getInitialState () {
    return {
      invoice: null
    }
```