# React-Router小结

1.简介

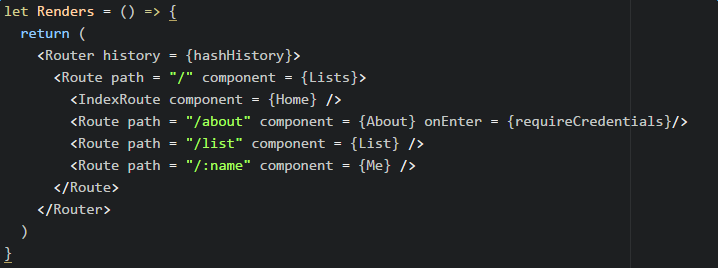
React-Router是由react官方维护的路由库，也是react唯一一个路由库，所以当你需要使用react的路由库时，你就可以使用这个优秀的库，使用之前，你要有个小小的准备工作。

2.起步

安装路由库  这就是使用node安装的依赖，这样你就可以开始你的react-router之旅了。

3.Router和Route组件

这两个组件是由react-router提供的，其中Router是最外层的包裹组件，Route是路由组件，基本用法如下：



他们的嵌套关系如上图所示，Router是最外层的包裹组件，他做的事就是包裹路由组件，将history传递给路由组件；而Route组件才是真正的路由组件，他负责路由切换时要显示哪一个组件。

4.Router组件详解

Router组件是一个包裹组件，这也是他的作用之一，他还有一个history属性，用来控制路由切换时是用hash值，还是正常的显示地址，不使用hash切换。

4.1 history属性

这个属性一共有三个属性值，hashHistory和browserHistory是前端控制地址显示的属性值，而createMemoryHistory属性值是控制服务端渲染的，与URL地址没有互动，这里就不做介绍。

4.1.1 hashHistory

这个属性值就是在我们切换路由时，以hash值得方式切换，地址显示的如下所示：



4.1.2 browserHistory

这个属性值在我们切换路由时地址栏显示的是正常的地址，如下所示：



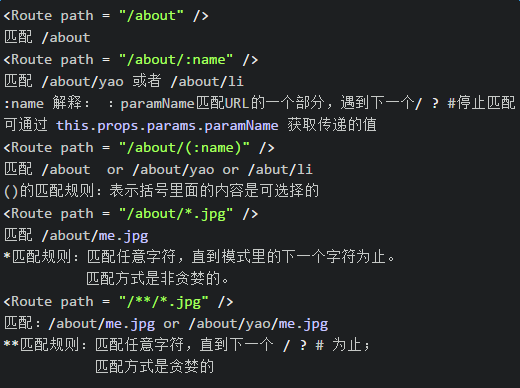
5.Ruote组件介绍

Route组件有常用两个属性path和component，path表示的是路由跳转将要跳转到哪个地址，component表示路由跳转到此地址时现实的组件。

5.1 path

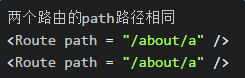
Path表示路径，他表示的是路由跳转的地址，你可以直接写上要跳转的地址，比如

这是最常用的，也是最容易理解的方式，下面将为你展示一张path写法的对照表，来看看路径的各种写法：



上图就是path属性的匹配规则，这样书写的路径最终都能匹配到相应的路由组件，

但这种匹配的路由，如果两个路由匹配的同一个路径，则只有第一个路由会生效，看下面的代码演示：



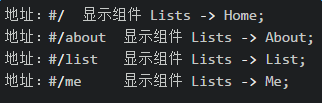
像上面这种情况，只有第一个路由会匹配，第二个是不会生效的，所以如果你要写有参数的路由路径，应该把带参数的路由写在整个路由的最下面。

5.2 component属性

这个属性就是你路由需要展示的组件。

6.路由嵌套

像上面的那个例子一样，路由可以嵌套使用，这样子显示路由组件的顺序是（以上面的例子为例）：



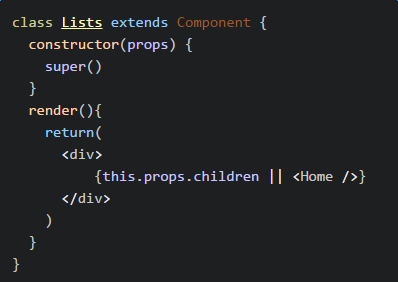
路由嵌套会先显示最外层的路由组件，再显示相应的组件，所以在Lists组建立你应该这样写：



你要用this.props.children来代替相应的路由组件，这样就可以达到路由切换，显示对应组件了。

7.IndexRoute组件

在上面的例子中，这个IndexRoute组件表示的是默认显示的组件，也就是路由一开始显示的组件，一般情况写，如果不用这个组件，我们在Lists中应该这样子写，是一开始的“/”路径可以由显示组件显示：



这样子在一开始就会默认显示home组件，但我们可以使用InedxRoute组件来简化这个过程，就像上面例子中的那样，在路径为“/”的时候，就会默认显示home组件。

8.Redirect组件

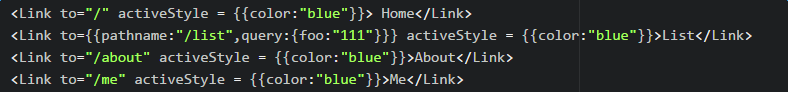
这个组件是用来进行重定向的，用法也很简单，看下面的例子：



这样就能实现路由跳转的重定向了。

9.Link组件

上面讲的都是怎样去定义路由组件，而Link相当于React自己定义的a连接，就是负责控制路由的跳转，看看下面的例子：



就像这样子，to属性对应的是你要跳转的路由的地址，跟Route组件的path的路径是对应的，activeStyle属性是这个Link组件被选中时显示的样式。有一点需要注意，Link组件要放在路由组件中才能生效，不如这个例子去，要将Link组件放在Lists组建中，也就是放在嵌套路由的最外面。所以在书写时，最好设置一个最外层的路由组件包裹其他的路由组件和Link组件。

9.1 Link 组件的to属性

这个属性比较特殊，他可以携带数据进行传递，比如上面的/list 路径携带这{foo：“111”}参数，他会直观的显示在地址栏上面：<http://localhost:3000/#/list?foo=111>，就像这样。

9.2 IndexLink 组件

这个组件是用来访问跟路由时用的，不常用，看下面的例子



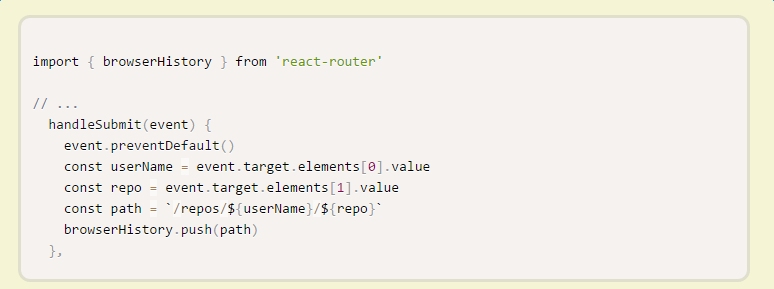
上图是对这个组件的完美展示。

10.表单处理

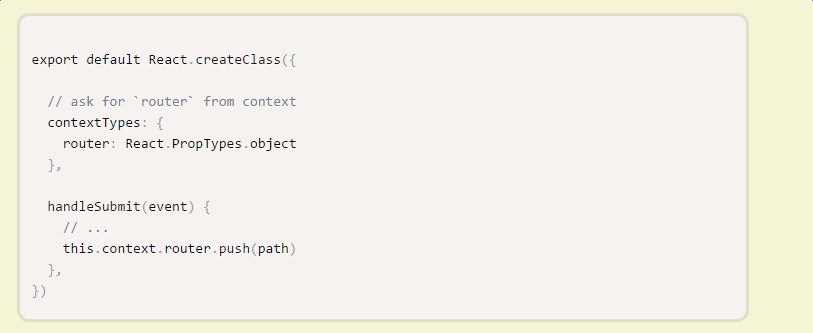
直接看下面的例子：



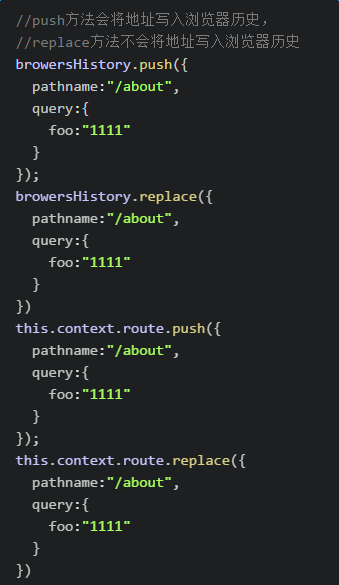
第一种方法：



第二种方法：



有一点需要提到的是，在上面的代码中，都会有一个push方法，用于路由的跳转切换，在这里，他还可以携带参数，看下面的代码：



Replace和push的区别也已写明，主要是调用了h5的history api，有兴趣的可以自己看一下。

11.钩子函数

11.1 onEnter

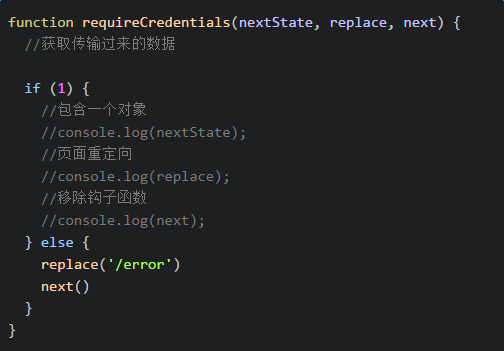
他的作用就是在进入路由时会触发，可以做一些参数处理等等，看代码：



就像绑定事件一样将事件绑定在onEnter上面。

11.2 onEnter的事件处理函数

直接上函数：



这个函数的参数有三个，下面将一一做介绍，来看看他们代表什么。

nextState：他说表示的是进入的这个路由的数据，包括地址，参数

replace：跟上面提到的replace用法一致，用来进行路由重定向

next：这个是在钩子函数执行后，将其移除，防止重复执行

11.3 Leave钩子函数

这个的用法相对复杂一些，他用到了React中的context

看代码：



上面这个例子中，Route组件me中的this.context.router.setRouteLeaveHook()方法就是用来定义离开时的函数，而routreWillLeave方法就是要执行的函数，参数nextLocation表示的是切换后的路由的地址，数据一类的参数，这个方法如果return一个字符串，则会通过弹出框告诉用户将要从那个路由切换到那个路由，如果返回值是一个false则会组织路由跳转，还在当前路由。