React-router 中学习笔记

参考网址：http://www.mrfront.com/2016/12/23/react-router-tutorial-part2/

1．为 Link 组件设置触发样式

为 index.html 添加样式

对 Link 组件使用 activeClassName>

```
15                  <li><Link to='/about' activeClassName='active'>About</Link></li>
16                  <li><Link to='/repos' activeClassName='active'>Repos</Link></li>
```

在 index.html 中添加 index.css

```
6     <link rel="stylesheet" type="text/css" href="index.css" />
```

Index.css 的内容：

```
1  .active {
2      color: red;
3  }
```

实现的效果如下：

如果是在 jsx 文件中要引入 css 文件的话，要使用 require(‘./style.css’)

Webpack-dev-server 的热替换功能不能作用于 index.html

2．将导航链接单独提取为一个组件

这样就不用担心因为 activeClassName 到处存在而难以维护

在 modules 中实现 NavLink.js

```
import React from 'react';
import {Link} from 'react-router'

export default class NavLink extends React.Component {
    constructor(props) {
        super(props);
    }
    // 这里的…this.props 点点点，使用了延展操作符（spread operator），它可以传递一组属性键值对给<Link/>
    render() {
        return (
            <Link {...this.props} activeClassName='active'/>
        );
    }
};
```

...传递属性键值对{activeClassName:active}给 this.props

将 App.js 中的<Link>改为<NavLink>即可

```
<ul role='nav'>
    <li><NavLink to='/about'>About</NavLink></li>
    <li><NavLink to='/repos'>Repos</NavLink></li>
</ul>
```

3.Url 参数

添加 modules/Repo.js

```
import React from 'react';

export default class Repo extends React.Component {
    constructor(props) {
        super(props);
    }
    render() {
        return (
            <div>
                <h2>{this.props.params.repoName}</h2>
            </div>
        );
    }
};
```

修改 index.js,path 属性指定路由的匹配规则

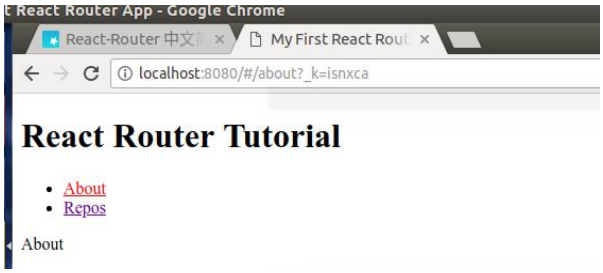
```
<Route path='/' component={App}>
    <Route path='/about' component={About} />
    <Route path='/repos' component={Repos} />
    {/* 添加Repo路由 */}
    <Route path='/repos/:userName/:repoName' component={Repo} />
</Route>
```

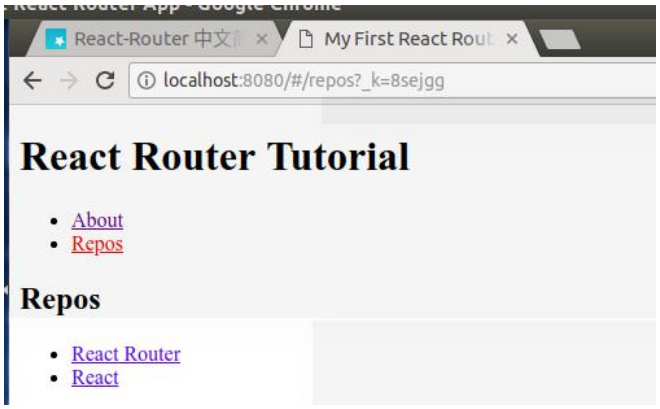
在 Repos.js 中添加链接，并且要引入 Link

```
<div>
    <h2>Repos</h2>
    {/* 添加链接 */}
    <ul>
        <li><Link to='/repos/reactjs/react-router'>React Router</Link></li>
        <li><Link to='/repos/facebook/react'>React Router</Link></li>
    </ul>
</div>
```

```
import {Link} from 'react-router';
```

npm start,在浏览器打开：localhost:8080





可以看到 `Repo` 组件的内容为：react-router，正是通过 `this.props.params.repoName` 属性获取到路由匹配规则 `/repos/:userName/:repoName` 中参数 `:repoName` 的具体值，即 URL 中 `/repos/reactjs/react-router` 的 **react-router** 字段。

4. 多层嵌套路由

与之前实现在 App.js 显示 About 类似
首先，将 index.js 总的 Repo 路由嵌套在 Repos 里面

```
22          <Route path='/' component={App}>
23              <Route path='/about' component={About} />
24              <Route path='/repos' component={Repos}>
25                  {/* 添加Repo路由,多层嵌套路由 */}
26                  <Route path='/repos/:userName/:repoName' component={Repo} />
27              </Route>
28          </Route>
```

然后将 Repos.js 里面的 Link 改为 NavLink，并且添加 this.props.children

```
15                  <ul>
16                      <li><NavLink to='/repos/reactjs/react-router'>React Router</NavLink></li>
17                      <li><NavLink to='/repos/facebook/react'>React</NavLink></li>
18                  </ul>
19
20                  <div>{this.props.children}</div>
```

运行结果如下：

5.indexRoute 组件

为了解决当子组件为空的时候，没有显示内容的情况，我们可以使用{this.props.children || <Home/>}使得默认显示 Home 组件的内容

在 modules 中添加 Home.js

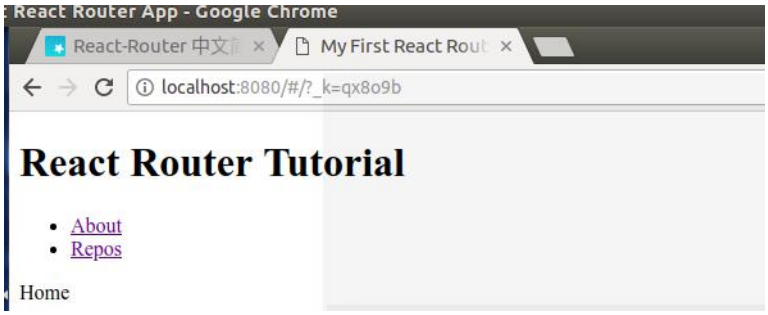
```
import React from 'react'

export default class About extends React.Component {
    constructor(props) {
        super(props);
    }
    render() {
        return (
            <div>Home</div>
        );
    }
};
```

虽然这样能够实现所要的效果，但是应该将 Home 放到路由中

修改 index.js 如下：将 IndexRoute 组件作为首页路由

```
        <Route path='/' component={App}>
            {/* 添加IndexRoute组件 */}
            <IndexRoute component={Home} />
            <Route path='/about' component={About} />
            <Route path='/repos' component={Repos}>
                {/* 添加Repo路由,多层嵌套路由 */}
```

执行结果为：



6．IndexLink 组件

在 App.js 中添加 Home 链接

```
                <ul role='nav'>
                    <li><NavLink to='/'>Home</NavLink></li>
```

为了解决 Home 始终处于激活状态的问题（子路由处于激活状态时，父路由也处于激活状态）改用 IndexLink 组件，修改 App.js 如下：

```
                <ul role='nav'>
                    <li><IndexLink to='/' activeClassName='active'>Home</IndexLink></li>
                    <li><NavLink to='/about'>About</NavLink></li>
                    <li><NavLink to='/repos'>Repos</NavLink></li>
                </ul>
```

```
2   import {Link, IndexLink} from 'react-router';
```

也可以使用另外一个方法来实现上面的效果：

```
<li><Link to="/" activeClassName="active" onlyActiveOnIndex={true}>Home</Link></li>
```

只需要在 NavLink 的基础上添加 onlyActiveOnIndex 属性即可

```
16                  <ul role='nav'>
17                      <li><NavLink to='/' onlyActiveOnIndex={true}>Home</NavLink></li>
18                      <li><NavLink to='/about'>About</NavLink></li>
19                      <li><NavLink to='/repos'>Repos</NavLink></li>
20                  </ul>
```

实现效果为：