# 一准备工作

# 二、路由基本使用

## 1、根目录的index文件：

import React from 'react';

import ReactDOM from 'react-dom';

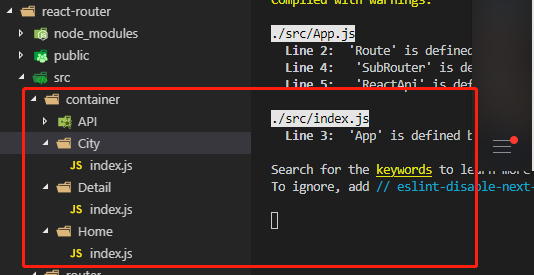
import AppRouter from './router/AppRouter';

ReactDOM.render(

<AppRouter/>, document.getElementById('root')

);

## 2、编写几个普通组件：



## 3、编写根组件：

import React from 'react'

import {Route,BrowserRouter } from 'react-router-dom'

import App from '../App'

import SubRouter from './SubRouter'

class AppRouter extends React.Component{

render(){

return (

<BrowserRouter>

<App>

<Route exact path='/' component={SubRouter}></Route>

{/\* //注意这里此时不能使用exact，因为子路由没有path='/' \*/}

</App>

</BrowserRouter>

)

}

}

export default AppRouter

## 4、编写App组件：

<li className='navlist'>

<Link to='/home'>主页</Link>

</li>

<li className='navlist'>

<Link to='/city'>城市</Link>

</li>

<li className='navlist'>

<Link to='/detail'>详情</Link>

</li>

<li className='navlist'>

<Link to='/routerApi'>react-API</Link>

</li>

</ul>

<div>

{this.props.children}

{/\* 这一步相当于编写App组件的子路由的多个Route，也就是SubRruter组件中的多个Route标签，所以这里可以像这样写,从而可以省去SubRruter组件：

<Switch>

<Route exact path='/' component={Home}></Route>

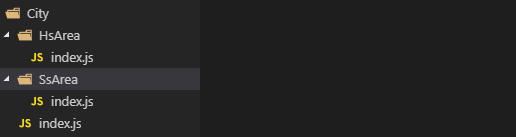
<Route exact path='/city' component={City}></Route>

<Route exact path='/detail' component={Detail}></Route>

</Switch> \*/}

# 三、嵌套路由，city组件嵌套子路由为例：

## 1、编写子路由组件



## 2、改写当前路由：

<Switch>

<Route exact path='/home' component={Home}></Route>

{/\* <Route exact path='/city' component={City}></Route> \*/}

<City path='/city' >

{/\* 注意这里City要使用path='/city' ，否则City后面的Route不会匹配 \*/}

<Route exact path='/city' component={CityRouter}></Route>

</City>

<Route exact path='/detail' component={Detail}></Route>

</Switch>

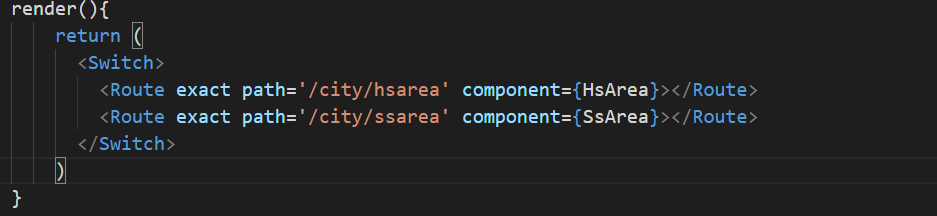
步骤：

2.1将subRouter中指向City组件的Route标签改为City标签，如上面箭头中。

2.2、将在City中编写Route标签，用于指定子路由

## 3、编写子路由：

CityRouter：



## 4:、编写city组件：

return(

<div>

City

<ul>

<li><Link to='/city/hsarea'>洪山区</Link></li>

<li><Link to='/city/ssarea'>松山区</Link></li>

</ul>

{this.props.children}

{/\* //这里表示子路由的Route标签 \*/}

</div>

# 四、路由传参

## 1、基本传参

## 2、传参方式

1、通过params

2、通过query:

3、通过state

## 3、动态传参：

1. 路由API：
2. Router

Router用于注册路由

1. Route
2. Link（路由跳转）
3. 路由嵌套：
4. 配置404页面：

配置路由表：

# React-router API

## Link

## NavLink

### 1、概念：

NavLink组件是另一个特殊的Link，可以在点击跳转页面的时候设置一些样式

### 2、属性：

#### ·1、activeClassName

#### ·2、activeStyle

#### ·3、exact

#### ·4、strict

#### ·5、isActive

#### ·6、location

#### ·7、aria-current

## Prompt（弹框）

山地人教程第5集

## MemoryRouter

## Redirect（重定向）

1. 介绍:

Redirect组件用于路由重定向，重定向就是路由默认跳转的Route

1. 属性
2. to（string/object）
3. push
4. from
5. exact
6. strict
7. sensitive

3、处理404

## Route

## Router

## StaticRouter

## Switch

## History

## Location

## Match

## matchPath

## withRouter