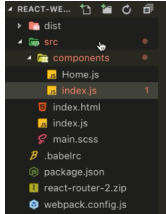


1. 安装指定版本react, react-dom, react-router, react-router-dom

```
bash-3.2# cnpm i react@16.7.0 react-dom@16.7.0 react-router@4.3.1 react-router-dom -S
```

2. 新建components文件夹, 文件夹新建Home.js, index.js文件, index.js文件设置接口暴露



```
src > components > index.js
1 export Home from './Home.js'
```

3. 在src下的index.js文件下引入Home, 注意引入书写方式的不同

```
src > index.js
1 // 入口文件
2 import React from 'react'; 7.9K (gzipped: 3.3K)
3 import ReactDOM from 'react-dom'
4 import { Home } from './components'
5 import style from './main.scss';
6
7 class Index extends React.Component{
8   render(){
9     return (
10       <div>
11         <Home/>
12       </div>
13     );
14   }
15 }
16
17 ReactDOM.render(<Index/>, document.getElementById('app'));
```

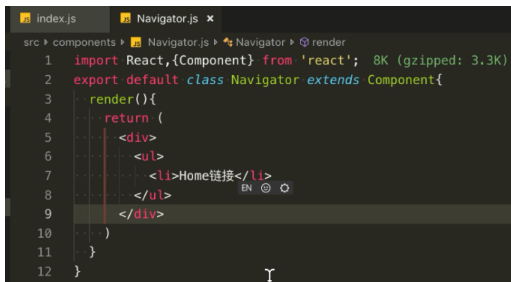
4. 在components文件夹下新建Navigator.js文件(导航文件, 放置链接), 在components文件夹下的index.js设置接口暴露

```
src > components > index.js
1 export Home from './Home.js'
2 export Navigator from './Navigator.js'
```

5. 在src下的index.js文件下引入Navigator, 渲染

```
src > index.js
1 // 入口文件
2 import React from 'react'; 7.9K (gzipped: 3.3K)
3 import ReactDOM from 'react-dom'
4 import { Home, Navigator } from './components'
5 import style from './main.scss';
6
7 class Index extends React.Component{
8   render(){
9     return (
10       <div>
11         <Navigator/>
12         <Home/>
13       </div>
14     );
15   }
16 }
17
18 ReactDOM.render(<Index/>, document.getElementById('app'));
```

Navigator.js文件放置Home链接

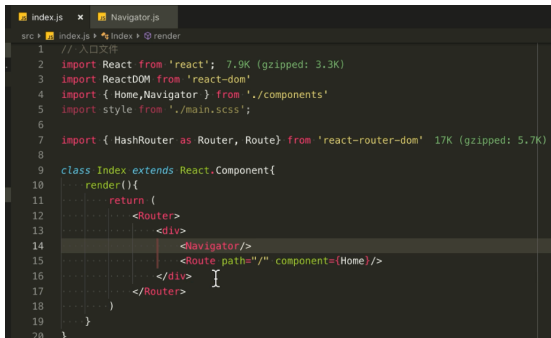


```

src > components > Navigator.js > Navigator > render
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 export default class Navigator extends Component{
3
4   render(){
5     return (
6       <div>
7         <ul>
8           <li>Home链接</li>
9         </ul>
10      </div>
11    )
12  }

```

6. 在src下的index. js文件（首页）下从react-router-dom下引入HashRouter, Route, Router 标签渲染整个路由表，Route渲染当前路由

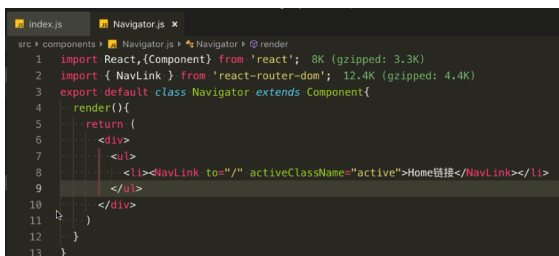


```

src > index.js > Index > render
1 // 入口文件
2 import React from 'react'; 7.9K (gzipped: 3.3K)
3 import ReactDOM from 'react-dom'
4 import { Home,Navigator } from './components'
5 import style from './main.scss';
6
7 import { HashRouter as Router, Route } from 'react-router-dom' 17K (gzipped: 5.7K)
8
9 class Index extends React.Component{
10   render(){
11     return (
12       <Router>
13         <div>
14           <Navigator/>
15           <Route path="/" component={Home}/>
16         </div>
17       </Router>
18     )
19   }
20 }

```

7. Navigator. js文件引入NavLink, 设置Home高亮显示（activeClassName）



```

src > components > Navigator.js > Navigator > render
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 import { NavLink } from 'react-router-dom'; 12.4K (gzipped: 4.4K)
3 export default class Navigator extends Component{
4   render(){
5     return (
6       <div>
7         <ul>
8           <li><NavLink to="/" activeClassName="active">Home链接</NavLink></li>
9         </ul>
10      </div>
11    )
12  }
13 }

```

8. 在components文件夹下新建BasicRouting. js文件, 在components文件夹下的index. js设置接口暴露, 在src下的index. js文件下引入BasicRouting, Route渲染当前路由

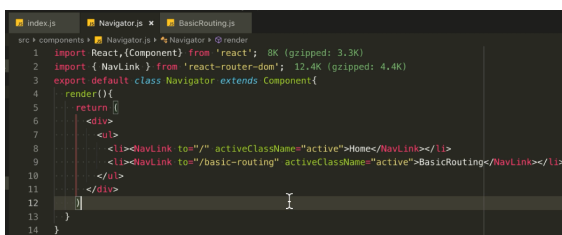


```

1 // 入口文件
2 import React from 'react'; 7.9K (gzipped: 3.3K)
3 import ReactDOM from 'react-dom'
4 import { Home,Navigator,BasicRouting } from './components'
5 import style from './main.scss';
6
7 import { HashRouter as Router, Route } from 'react-router-dom' 17K (gzipped: 5.7K)
8
9 class Index extends React.Component{
10   render(){
11     return (
12       <Router>
13         <div>
14           <Navigator/>
15           <Route path="/" component={Home}/>
16           <Route path="/basic-routing" component={BasicRouting}/>
17         </div>
18       </Router>
19     )
20   }
21 }

```

Navigator. js文件设置basic-routing链接

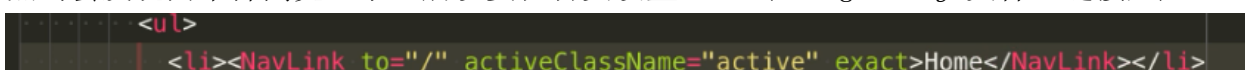


```

src > components > Navigator.js > Navigator > render
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 import { NavLink } from 'react-router-dom'; 12.4K (gzipped: 4.4K)
3 export default class Navigator extends Component{
4   render(){
5     return (
6       <div>
7         <ul>
8           <li><NavLink to="/" activeClassName="active">Home</NavLink></li>
9           <li><NavLink to="/basic-routing" activeClassName="active">BasicRouting</NavLink></li>
10        </ul>
11      </div>
12    )
13  }
14 }

```

点击会发现两个都高亮显示，所以要在首页设置exact (Navigator. js文件，链接处)



```

<li><NavLink to="/" activeClassName="active" exact>Home</NavLink></li>

```

同样的，下面内容既显示Home又显示BasicRouting, 也可以在首页设置exact（src下的index.js文件，路由处）

```
<Route path="/" component={Home} exact|/>
```

这里的Route可以理解为占位显示

9. BasicRouting文件引入NavLink，设置链接

```
src > components > BasicRouting.js > BasicRouting > render
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 import {NavLink} from 'react-router-dom' 12.4K (gzipped: 4.4K)
3 export default class BasicRouting extends Component{
4   render(){
5     console.log(this.props)
6     return (
7       <div>
8         <ul>
9           <li><NavLink to={this.props.match.url + '/level1'}>Level 1</NavLink></li>
10          <li><NavLink to={this.props.match.url + '/level2'}>Level 2</NavLink></li>
11          <li><NavLink to={this.props.match.url + '/level3'}>Level 3</NavLink></li>
12        </ul>
13      </div>
14    )
15  }
16 }
17 }
```

10. 路由嵌套，在components文件夹下新建Content.js文件, 在components文件夹下的index.js设置接口暴露, 在BasicRouting文件下引入Content (因为要在点击BasicRouting时显示，即子组件内容)，引入Route（当前路由）

```
src > components > BasicRouting.js > BasicRouting > render
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 import {NavLink,Route} from 'react-router-dom' 12.5K (gzipped: 4.4K)
3 import {Content} from '../components'
4
5 export default class BasicRouting extends Component{
6   render(){
7     console.log(this.props)
8     return (
9       <div>
10        <ul>
11          <li><NavLink to={this.props.match.url + '/level1'}>Level 1</NavLink></li>
12          <li><NavLink to={this.props.match.url + '/level2'}>Level 2</NavLink></li>
13          <li><NavLink to={this.props.match.url + '/level3'}>Level 3</NavLink></li>
14        </ul>
15        <Route path={` ${this.props.match.url} /:level` } component={Content}/>
16      </div>
17    )
18  }
19 }
20 }
```

这里Route既是路由，又是占位显示，：level参数，设参在路由

```
</ul>
... { /* Route既是路由又是占位渲染显示 */ }
... <Route path={` ${this.props.match.url} /:level` } component={Content}/>
```

传参在地址

```
<ul>
  { /* this.props.match.url 动态地址获取 */ }
  <li><NavLink to={this.props.match.url + '/level1'}>Level 1</NavLink></li>
  <li><NavLink to={this.props.match.url + '/level2'}>Level 2</NavLink></li>
  <li><NavLink to={this.props.match.url + '/level3'}>Level 3</NavLink></li>
</ul>
```

接参在组件

```
index.js  Navigator.js  BasicRouting.js  Content.js
src > components > Content.js > Content > render
1  import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2  export default class Content extends Component{
3    render(){
4      console.log(this.props)
5      const { params } = this.props.match;
6      const location = this.props.location;
7      return (
8        <div>
9          Content
10         <hr/>
11         Params: { JSON.stringify(params) }
12         <br/>
13         Location: {JSON.stringify(location)}
14       </div>
15     )
16   }
17 }
```

11. 离开页面时提示文字

在components文件夹下新建Blocking.js文件,在components文件夹下的index.js设置接口暴露,在src下的index.js文件下引入BasicRouting,Route渲染当前路由,Navigator.js文件设置blocking链接,Blocking.js文件引入Prompt

```
index.js  Navigator.js  Blocking.js  BasicRouting.js  Content.js
src > components > Blocking.js > Blocking > render
1  import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2  import { Prompt } from 'react-router-dom' 3.5K (gzipped: 1.5K)
3  export default class Blocking extends Component{
4    render(){
5      return (
6        <div>
7          Blocking
8        </div>
9      )
10     <Prompt message="你确认要离开该页面吗?"/>
11   }
12 }
13 }
```

12. 找不到地址（地址错误），显示NoMatch，相当于Vue里的重定向（redirect）

在components文件夹下新建Miss.js文件,在components文件夹下的index.js设置接口暴露,在src下的index.js文件下引入Miss,Route渲染当前路由,Navigator.js文件设置Miss链接Miss.js文件引入Route,NavLink,Content,设置,level2地址错误

```
index.js  Navigator.js  Blocking.js  Miss.js  Content.js
src > components > Miss.js > Miss > render
1  import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2
3  import {Route,NavLink} from 'react-router-dom' 12.5K (gzipped: 4.4K)
4  import {Content} from '../components'
5
6  export default class Miss extends Component{
7    render(){
8      return (
9        <div>
10         Miss
11         <hr/>
12         <ul>
13           <li><NavLink to={this.props.match.url + '/abc/level1'}>Level 1</NavLink></li>
14           <li><NavLink to={this.props.match.url + '/level2'}>Level 2</NavLink></li>
15           <li><NavLink to={this.props.match.url + '/abc/level3'}>Level 3</NavLink></li>
16         </ul>
17
18         <Route path={` ${this.props.match.url}/abc/:level`} component={Content}/>
19       </div>
20     )
21   }
22 }
```

在components文件夹下新建NoMatch.js文件,在components文件夹下的index.js设置接口暴露,在Miss文件下引入NoMatch,Route渲染当前路由,注意这里不写path路径

```
src > components > Miss.js > render
7 render(){
8   return (
9     <div>
10       Miss
11       <hr/>
12       <ul>
13         <li><NavLink to={this.props.match.url + '/abc/level1'}>Level 1</NavLink></li>
14         <li><NavLink to={this.props.match.url + '/level2'}>Level 2</NavLink></li>
15         <li><NavLink to={this.props.match.url + '/abc/level3'}>Level 3</NavLink></li>
16       </ul>
17       <Route path={`/${this.props.match.url}/abc/:level`} component={Content}/>
18       <Route component={NoMatch}/>
19     </div>
20   )
}
```

这里会发现无论地址正确还是错误都会显示NoMatch，所以再在Miss文件引入Switch

```
src > components > Miss.js > ...
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2
3 import {Route,NavLink,Switch} from 'react-router-dom' Calculating...
4 import {Content,NoMatch} from '../components'
```

设置选择判断，NoMatch放在最后面

```
/* Switch是选择判断，如果哪个路由已经匹配，将不继续之后路由的匹配操作 */
<Switch>
  <Route path={`/${this.props.match.url}/abc/:level`} component={Content}/>
  <Route component={NoMatch}/>
</Switch>
```

同理，我们也可以在src下的index.js文件下引入Switch, NoMatch

```
src > index.js > Index
1 // 入口文件
2 import React from 'react'; 7.9K (gzipped: 3.3K)
3 import ReactDOM from 'react-dom'
4 import { Home,Navigator,BasicRouting,Blocking,Miss,NoMatch } from './components'
5 import style from './main.scss';
6
7 import { HashRouter as Router, Route,Switch} from 'react-router-dom' 18.8K (gzipped: 5.9K)
```

设置

```
<Switch>
  <Route path="/" component={Home} exact/>
  <Route path="/basic-routing" component={BasicRouting}/>
  <Route path="/blocking" component={Blocking}/>
  <Route path="/miss" component={Miss}/>
  <Route component={NoMatch}/>
</Switch>
```

13. url地址参数传递

在components文件夹下新建QueryParams.js文件, 在components文件夹下的index.js设置接口暴露, 在src下的index.js文件下引入QueryParams, Route渲染当前路由, Navigator.js文件设置QueryParams链接

QueryParams.js文件

```
src > components > JS QueryParams.js > QueryParams > render
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 import {NavLink,Route,Switch} from 'react-router-dom'
3 import {Content,NoMatch} from '../components'
4
5 export default class QueryParams extends Component{
6   render(){
7
8     const {location,match}=this.props;
9     return(
10      <div>
11        <QueryParams
12        <hr/>
13        <ul>
14
15          <li><NavLink to={
16            {pathname:match.url+'/'level1',
17             search:'?abc=123'}
18          }>Level 1</NavLink></li>
19          <li><NavLink to={
20            {pathname:match.url+'/'level2',
21             search:'?abc=123&xyz=456'}
22          }>Level 2</NavLink></li>
23          <li><NavLink to={match.url+'/'level3?abc=123&xyz=456'}>Level 3</NavLink></li>
24        </ul>
25      </div>
26    );
27  }
28 }
29
30 /* Switch是选择判断, 如果哪个路由已经匹配, 将不继续之后路由的匹配操作 */
31 <Switch>
32   <Route path={`/${match.url}/${level}` component={Content} />
33   <Route component={NoMatch} />
34 </Switch>
35
36 </div>
37 )
38 }
39 }
```

14. 递归 (Recursive)

在components文件夹下新建Recursive.js文件, 在components文件夹下的index.js设置接口暴露, 在src下的index.js文件下引入Recursive, Route渲染当前路由, Navigator.js文件设置Recursive链接

Recursive.js文件

```
src > components > JS Recursive.js > Recursive
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 import {NavLink,Route} from 'react-router-dom' 31.3K (gzipped: 9K)
3
4
5 export default class Recursive extends Component{
6   render(){
7     return(
8       <div>
9         <Recursive
10         <hr/>
11         <ul>
12
13           <li><NavLink to={this.props.match.url+'/'level1'}>Level 1</NavLink></li>
14           <li><NavLink to={this.props.match.url+'/'level2'}>Level 2</NavLink></li>
15           <li><NavLink to={this.props.match.url+'/'level3'}>Level 3</NavLink></li>
16         </ul>
17         <Route path={`/${this.props.match.url}/${level}` component={Recursive} />
18       </div>
19     );
20   }
21 }
```

15. 路由配置

在src下新建router-config.js文件

```
ex.js ...components JS index.js src JS QueryParams.js JS Recursive.js JS router-config.js JS Navigator.js
JS router-config.js > <unknown>
1 import React from 'react'
2 import {Home,BasicRouting,Blocking,Miss,QueryParams,Recursive} from './components';
3
4 module.exports=[
5   {
6     path: '/',
7     component:Home,
8     exact:true
9   },
10  {
11    path: '/basic-routing',
12    component:BasicRouting
13  },
14  {
15    path: '/blocking',
16    component:Blocking
17  },
18  {
19    path: '/miss',
20    component:Miss
21  },
22  {
23    path: '/query-params',
24    component:QueryParams
25  },
26  {
27    path: '/recursive',
28    component:Recursive
29  }
30 ]
```

src下index. js文件

```
index.js ...components JS index.js src JS QueryParams.js JS Recursive.js JS router-config.js JS Navigator.js JS Blocking.js JS Miss.js JS N
src JS index.js > index > @render
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import style from './main.css';
4 import {Navigator,NoMatch} from './components';
5 import routes from './router-config'
6
7 import {HashRouter as Router,Route,Switch} from 'react-router-dom'
8 class Index extends React.Component{
9   render(){
10     return(
11       <Router>
12         <div>
13           <Navigator/>
14           <Switch>
15             {
16               routes.map(({route,index})=>{
17                 return <Route key={index} path={route.path} component={route.component} exact={route.exact}/>
18               })
19             }
20             <Route component={NoMatch} />
21           </Switch>
22         </div>
23       </Router>
24     )
25   }
26 }
27
28 ReactDOM.render(<Index/>,document.getElementById('app'))
```

16. 程序式导航

```
JS Home.js JS BasicRouting.js
src components JS Home.js > Home > handleClick
1 import React,{Component} from 'react'; 8K (gzipped: 3.3K)
2 export default class Home extends Component{
3
4   handleClick=()=>{
5     this.props.history.push('./basic-routing')
6   }
7
8   render(){
9     return(
10       <div>
11         Home
12         <button onClick={this.handleClick}>Goto</button>
13       </div>
14     )
15   }
16 }
```

