爱创课堂前端培训

# ReactJS

第4天课堂笔记（本课程共4天）

讲师：张容铭

日期：2018年6月29日

张容铭老师

微博：@张容铭\_YYQH

QQ : 286031482

E-mail : yuye\_qinghe@qq.com

爱创课堂官网 ：[www.icketang.com](http://www.icketang.com)

目录

[名称 1](#_Toc8043)

[一、路由 3](#_Toc24183)

[1.1使用路由 3](#_Toc4161)

[1.2路由模块 3](#_Toc3400)

[1.3路由信息 4](#_Toc369)

[1.4路由中使用redux 4](#_Toc15338)

[二、混合 7](#_Toc32146)

[三、项目实战 8](#_Toc917)

[3.1路由 8](#_Toc30220)

# 一、路由

我们使用路由的目的是为了实现单页面应用程序（SPA），

一个完整的url分成几个部分

协议，域名，端口号，路径，文件，搜索词，哈希

只有hash的改变，不会导致浏览器端想服务器端发送新的请求，所以前端的路由是基于hash实现的

在React中，版本更新变化是比较大的，因此每一个版本都有一套对应的路由

React为了实现特点3（多端适配），因此将路由做了拆分，来适应各个端

https://github.com/ReactTraining/react-router

## 1.1使用路由

在react中使用路由分三步

第一步 在应用程序组件中，定义路由渲染容器，通过Switch组件定义

在里面通过Route组件定义每一条规则

name表示路由名称

component表示渲染的组件

path 表示路由规则

通过/将路由分成两个部分

默认是不变的（静态路由）（一对一的）

我们可以通过在模块名称前面添加:定义动态路由（一对多的）

可以通过?定义query路由等等

如果想匹配所有的，我们用\*表示（由于\*匹配的比较广，所以我们通常定义在最后面）

第二步 选择路由渲染策略，渲染应用程序组件

React16版本提供了两种侧路

一种是基于path的改变实现的路由，叫BrowserRouter，需要后端配合

一种是基于hash的改变实现的路由，叫HashRouter，前端独立完成

第三步 渲染第二步得到的结果（路由规则）

## 1.2路由模块

BrowserRouter 基于path的改变，实现的路由策略

HashRouter 基于hash的改变，实现的路由策略

Link 路由导航组件（实现页面切换的，默认渲染成a标签）

to属性定义地址

在hash策略中，与a标签的差别是

a标签要写#

Link直接写路径

MemoryRouter 内存路由策略

NavLink 导航链接

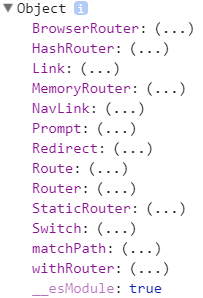
Redirect 重定向组件

from 定义来源页

to 重定向的页面

Route 定义每一条规则的

Switch 路由容器



## 1.3路由信息

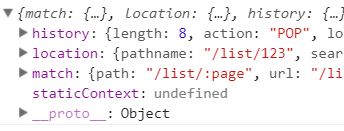
路由渲染的组件会添加一些信息，我们可以通过组件的属性数据获取他们

路由子组件是不具备这些信息的，想拥有这些信息，我们可以传递属性数据

history react路由内部维护的history对象

location router路由模拟location对象，存储自身的路由信息（hash模块的切割）

match 路由匹配的数据



## 1.4路由中使用redux

我们通过Provider组件渲染页面DealApp，为了传递store，

路由通过HashRouter|BrowserRouter渲染组件

1 所以我们HashRouter|BrowserRouter放在Provider中即可

2 在HashRouter|BrowserRouter组件中，通过Route渲染DealApp组件

想绑定路由调度者，我们可以使用redxu提供的combineReducers方法实现绑定

|  |
| --- |
| 1. // 引入库 2. import React, { Component } from 'react'; 3. import { render } from 'react-dom'; 4. import { Switch, Route, HashRouter, Link, Redirect, BrowserRouter } from 'react-router-dom'; 5. // 引入redex 6. import { createStore, combineReducers } from 'redux'; 7. // 引入react-redux 8. import { connect, Provider } from 'react-redux'; 9. // 引入处理路由的调度者 10. import { routerReducer } from 'react-router-redux'; 11. // 定义消息名称 12. const ADD = 'ADD'; 13. // 定义调度者 14. function reducer(state = 0, action) { 15. if (action.type === ADD) { 16. return state += action.num 17. } 18. return state; 19. } 20. // 创建store 21. let store = createStore( 22. combineReducers({ 23. num: reducer, 24. routerReducer 25. }) 26. ); 27. class ShowChild extends Component { 28. render() { 29. // console.log(this.props) 30. return <h3>show: {this.props.state}</h3> 31. } 32. } 33. class ChangeChild extends Component { 34. render() { 35. // console.log(this.props) 36. return <button onClick={e => this.props.dispatch({ type: ADD, num: 10 })}>点击改变</button> 37. } 38. } 39. // 定义组件 40. class App extends Component { 41. // 渲染 42. render() { 43. // console.log(this.props) 44. let {state, dispatch} = this.props; 45. return ( 46. <div> 47. <h2>应用程序组件</h2> 48. <ShowChild state={state.num}></ShowChild> 49. <ChangeChild dispatch={dispatch}></ChangeChild> 50. {/\*第一步定义路由渲染位置\*/} 51. <Switch> 52. {/\*列表页，页码是动态改变的\*/} 53. <Route path="/list/:page" component={DealList}></Route> 54. {/\*详情页,商品的id是可变的\*/} 55. <Route path="/detail/:id" component={Detail}></Route> 56. {/\*默认路由，输入其他的，我们都想进入首页\*/} 57. <Route path="\*" component={Home}></Route> 58. </Switch> 59. </div> 60. ) 61. } 62. } 63. // home 64. class Home extends Component { 65. // 渲染 66. render() { 67. return <h1>home page</h1> 68. } 69. } 70. // list 71. class List extends Component { 72. // 渲染 73. render() { 74. // 将页码渲染出来 75. console.log(123, this.props) 76. return ( 77. <div> 78. <h1>list page : {this.props.match.params.page}</h1> 79. <Demo match={this.props.match}></Demo> 80. </div> 81. ) 82. } 83. } 84. class Demo extends Component { 85. // 渲染 86. render() { 87. // 将页码渲染出来 88. console.log('demo', this.props) 89. return <h3>demo page </h3> 90. } 91. } 92. // detail 93. class Detail extends Component { 94. // 渲染 95. render() { 96. return <h1>detail page</h1> 97. } 98. } 99. // 对应用程序APP拓展store属性和方法 100. function mapStateToProps(state) { 101. return { state } 102. } 103. function mapDispatchToProps(dispatch) { 104. return { dispatch } 105. } 106. // 为应用程序组件APP拓展 107. let DealApp = connect(mapStateToProps, mapDispatchToProps)(App) 108. // 为List添加store中是信息 109. let DealList = connect(mapStateToProps, mapDispatchToProps)(List) 110. // 第二步 定义渲染策略 111. let routes = ( 112. <HashRouter> 113. <Route path="/" component={DealApp}></Route> 114. {/\*<DealApp></DealApp>\*/} 115. </HashRouter> 116. ) 117. // 渲染路由规则 118. render( 119. // 渲染应用程序, 为应用程序提供store 120. <Provider store={store}> 121. {routes} 122. </Provider> 123. , app) |

# 二、混合

混合就是为了复用组件的方法，ES5中通过mixins继承混合对象（类似多继承）

ES6中实现了继承，就不需要mixins，直接通过extend继承混合类

ES6中的混合

1 混合类继承组件基类

2 组件类继承混合类

|  |
| --- |
| 1. // 引入库 2. import React, { Component } from 'react'; 3. import { render } from 'react-dom'; 4. // 定义混合类 5. class ListMixin extends Component { 6. // 渲染列表 7. createList() { 8. return this.props.data.map((item, index) => { 9. return <li key={index}>{item}</li> 10. }) 11. } 12. } 13. // 定义首页组件 14. // 组件类继承混合类 15. class Home extends ListMixin { 16. render() { 17. return ( 18. <div> 19. <h2>Home page</h2> 20. <ul>{this.createList()}</ul> 21. </div> 22. ) 23. } 24. } 25. // 定义list组件 26. class List extends ListMixin { 27. render() { 28. return ( 29. <div> 30. <h2>list page</h2> 31. <ul>{this.createList()}</ul> 32. </div> 33. ) 34. } 35. } 36. render(<Home data={["不能去了！九寨沟景区7月1日起关闭", "书记：若你丈夫对你不好 我找人做了他", "现场画面：长征火箭残骸坠落贵州爆炸"]}></Home>, app) 37. render(<List data={["比利时1-0胜英格兰进死亡半区！", "世界杯第一欢乐男！踢球自虐 这一幕...", "小组赛最佳阵:C罗携凯恩 魔笛库鸟领衔", "中国男足饮食曝光 难怪进不了世界杯！"]}></List>, app2) |

# 三、项目实战

技术架构

react， react-dom, react-router, less, webpack, axios

目录部署

img 存储项目图片

data 存储接口数据

modules 所有模块

app 应用程序组件

components 组件目录

page 所有页面组件目录

router 路由模块

bootstrap.jsx 启动页面

webpack.config.js 配置文件

index.html 首页

## 3.1路由

输入/detail/10 进入详情页 => /detail/:id

输入/comments/2 进入评论页面 => /comments/:id

其他的进入首页 默认路由

|  |
| --- |
| 1. // 引入react 2. import React, { Component } from 'react'; 3. // 引入路由模块 4. import { Switch, Route } from 'react-router-dom'; 5. // 引入页面组件 6. import Home from '../pages/home/home.jsx'; 7. import Detail from '../pages/detail/detail.jsx'; 8. import Comments from '../pages/comments/comments.jsx'; 9. // 引入header组件 10. import Header from '../components/header/header.jsx'; 11. // 全局样式 12. import './app.less'; 13. // 定义组件 14. export default class App extends Component { 15. render() { 16. return ( 17. <div> 18. <Header></Header> 19. {/\* 第一步 定义渲染位置 \*/} 20. <Switch> 21. {/\*详情页\*/} 22. <Route path="/detail/:id" component={Detail}></Route> 23. {/\*评论页\*/} 24. <Route path="/comments/:id" component={Comments}></Route> 25. {/\*首页\*/} 26. <Route path="\*" component={Home}></Route> 27. </Switch> 28. </div> 29. ) 30. } 31. } |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |