爱创课堂前端培训

# React 16

第5天课堂笔记（本课程共5天）

班级：北京前端训练营16期

讲师：张容铭

日期：2018年11月11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

[React 16 1](#_Toc18566)

[复习 3](#_Toc25937)

[一、element-react 4](#_Toc28950)

[1.1体验element-react 4](#_Toc22216)

[二、create-react-app 4](#_Toc15457)

[2.1目录部署 5](#_Toc29307)

[三、单元测试 6](#_Toc23212)

[3.1启动单测 6](#_Toc1294)

[3.2 describe 7](#_Toc5558)

[3.3 it 7](#_Toc19016)

[3.4 expect 7](#_Toc30498)

[3.5预言方法 7](#_Toc22093)

[3.6生命周期方法 7](#_Toc30549)

[3.7测试虚拟DOM 8](#_Toc18839)

[3.8项目发布 8](#_Toc8178)

[3.9输出webpack配置 8](#_Toc11752)

[下午复习 10](#_Toc10149)

[四、项目实战 11](#_Toc2945)

[4.1技术架构 11](#_Toc30942)

[4.2目录架构 11](#_Toc360)

[4.3路由 11](#_Toc25498)

[4.4 react拓展 11](#_Toc16514)

# 复习

redux 特点：1 单一数据源，2 state是只读的， 3 函数式编程

action: 对象， type

reducer：函数 state, action, return

store: 存储数据，createStore

dispatch, subscribe, getState

观察者模式

redux解决通信

connect(mapStateToProps, mapDispatchToProps)(App)

Provider store

后代组件：1 传递属性， 2 拓展组件

路由 1 Swtich， Route[name, path, component] 2 HashRouter|BrowserRouter 3 render

默认路由 \*

Redirect from to

路由数据 histroy， location， match

路由导航 Link to

路由中使用redux

1 HashRouter Route 2 Provider routes

路由reducer combineReducers({ key: value })

state: 只读的，reducer 对象

action 动态action

异步action applyMiddleware(thunkMiddleware)(createStore)

函数返回值参数是dispatch

antd-mobile

options { presets: [], plugins: [ [‘import’, { style: css, libraryName: antd-mobile }] ] }

组件

# 一、element-react

element-ui不仅仅提供了vue方面的组件，也提供了react，angular方向的组件，功能都是一样的。

但是在每个模块使用的组件库的名称是不一样的

如

vue => element-ui

react => element-react

我们在react中使用，就要安装element-react

npm install element-react

此时我们就可以在项目中使用了

## 1.1体验element-react

默认没有样式，我们要引入主题样式

npm install element-theme-default

注意：都是css文件，因此要配置css加载机

在主题中，使用了字体图标等字体文件，因此我们要通过webpack处理。

可以通过url-loader加载。

|  |
| --- |
| 1. import React, { Component } from 'react'; 2. import { render } from 'react-dom'; 3. // 使用element-react 4. import { Button, Alert, Tag } from 'element-react'; 5. // 引入样式 6. import 'element-theme-default' 7. // 定义组件 8. class App extends Component { 9. render() { 10. return ( 11. <div> 12. <h1>hello</h1> 13. <Button type="success">hello</Button> 14. <Button type="info">hello</Button> 15. <Button type="warning">hello</Button> 16. <Button type="danger">hello</Button> 17. <Alert title="爱创课堂" description="专业前端培训学校" closeText="hello"></Alert> 18. <Tag type="gray">标签二</Tag> 19. </div> 20. ) 21. } 22. } 23. // 渲染应用程序 24. render(<App></App>, app) |

# 二、create-react-app

vue为了简化我们创建项目，提供了vue-cli

react为了简化我们创建项目，提供了create-react-app

提供了create-react-app指令，所以要全局安装

npm install -g create-react-app

查看版本号 create-react-app -V



创建项目 create-react-app 项目名称(横向分割单词)

create-react-app也是通过yarn管理的，因此启动项目我们既可以使用npm run start指令，也可以使用yarn start

## 2.1目录部署

node\_modules 所有依赖模块集合

public 静态模块（通常不会被webpack处理）

src 开发目录（工作中99%的内容，都是在该目录中）

App.css 应用程序样式

App.js 应用程序脚本

App.test.js 应用程序的单元测试文件

index.css 全局样式文件

logo.svg react logo

registerSeviceWorker.js web work文件

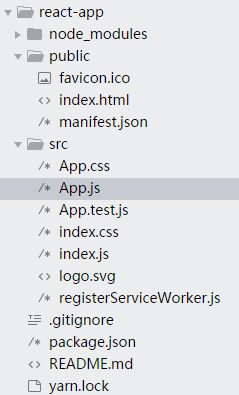
.gitignore git忽略的文件配置

.package.json 项目配置文件

README.md 介绍文件

yarn.lock yarn锁文件

注意：在create-react-app创建的项目中，组件文件的拓展名可以是.js也可以是.jsx



|  |
| --- |
| 1. yarn start |

# 三、单元测试

为了保证我们开发代码的质量，我们要做单元测试，也叫单测。

前端单测工具有很多，例如

angular6使用jesmine

react使用jest

...

create-react-app创建的项目，也内置了单测，内置的工具是jest工具

在项目中，测试文件有三类

1 \_\_test\_\_目录中的文件

2 .test.js|.test.jsx结尾的文件

3 .spec.js|.spec.jsx结尾的文件

测试文件命名规范

通常是与组件的名称一致的，例如 App.js的测试文件App.test.js

测试文件中要提供一些测试语句。

注意：jest单测工具，支持中文字符

## 3.1启动单测

yarn test 指令可以启动单测

单测运行的时候，会执行每一个单测语句，判断每一个单测结果

全部通过，单测就通过了

有一个没有通过，单测就没有通过

## 3.2 describe

用来描述整个测试的

第一个参数，就是测试描述（具有目的性）

第二个参数，是函数，在函数中我们书写单元测试

## 3.3 it

用来定义单测语句的

第一个参数，本次单测的描述（具有目的性）

第二个参数，是函数，在函数中，我们实现单测

## 3.4 expect

定义测试表达式

参数就是我们处理的逻辑

我们通过一些方法判断他的返回值，来达到预言的目的

## 3.5预言方法

toBe

相当于 ===

toEqual

与参数字面量是否相等

toMatch

与参数正则是否匹配

toContain

是否包含参数成员

toBeTruthy

转换成布尔值是否为真

toBeFalsy

转换成布尔值是否为假

...

## 3.6生命周期方法

beforeEach(fn)

每个单测执行前，执行的方法

afterEach(fn)

每个单测执行后，执行的方法

beforeAll(fn)

所有单测执行前

afterAll(fn)

所有单测执行后

## 3.7测试虚拟DOM

jest测试只能测试一些逻辑问题，但是react中，绝大部分时候，我们是在写虚拟DOM以及组件，如何测试它们呢，react提供了enzyme模块，提供了shallow方法，可以模拟创建一个组件供我们测试。

由于react版本更新迭代很快，因此enzyme也需要更新，后来采取类似babel模块的策略，

为每一个react的版本提供适配器模块。

例如react 16版本的适配器： enzyme-adapter-react-16,所以我们要安装这两个模块

yarn add enzyme enzyme-adapter-react-16

此时我们就可以使用了

我们在src/setupTests.js中定义适配器配置

configure({adpter: new Adapter()})

configure方法由enzyme提供

Adapter类就是enzyme-adapter-react-16模块

注意：工作中，建议大家，多测试逻辑部分。视图部分建议用浏览器工具测试

## 3.8项目发布

react会自动帮助我们的项目文件，实现打包，压缩，加指纹等性能优化的操作了。

## 3.9输出webpack配置

我们执行yarn eject之后，此时就将webpack的配置输出出来，是不可逆的，要慎用。

|  |
| --- |
| 1. import React from 'react'; 2. import ReactDOM from 'react-dom'; 3. import App from './App'; 4. // 引入测试虚拟DOM模块 5. import { shallow } from 'enzyme'; 6. // 引入文件 7. import { sum } from './util/util'; 8. // 整体描述 9. describe('我们进行单元测试', () => { 10. // 定义生命周期方法 11. // 模拟的组件 12. let wrapper; 13. // 每个单测执行前 14. beforeEach(() => { 15. // console.log('beforeEach') 16. wrapper = shallow(<App></App>) 17. }) 18. // // 每个单测执行后 19. afterEach(() => { 20. // console.log('afterEach') 21. // 每次测试完，将该模拟的组件注销掉 22. wrapper.unmount(); 23. }) 24. // // 所有单测执行前 25. // beforeAll(() => { 26. // console.log('before all') 27. // }) 28. // // 所有单测执行后 29. // afterAll(() => { 30. // console.log('after all') 31. // }) 32. // 定义单元测试 33. it('测试sum方法是否能够得到正确结果', () => { 34. // 执行表达式 35. expect(sum(2, 3)) 36. // 是否等于4 37. .toBe(5) 38. }) 39. it('测试字面量', () => { 40. expect({color: 'red'}) 41. // 是否字面量相等 42. .toEqual({color: 'red'}) 43. }) 44. it('正则测试', () => { 45. expect('hello') 46. // 是否以hello开头 47. .toMatch(/^hello/) 48. }) 49. it('数组是否包含某个成员', () => { 50. expect([1, 2, 3]) 51. // 是否包含2 52. .toContain(2) 53. }) 54. it('测试组件', () => { 55. // 模拟一个组件 56. // let wrapper = shallow(<App></App>); 57. // 是否包含h1元素 58. expect(wrapper.contains(<h1 className="App-title">Welcome to React</h1>)).toBe(true) 59. }) 60. it('是否可以找到某个组件', () => { 61. expect(wrapper.find('.App-header')) 62. // 是否具有，就看长度是几 63. .toHaveLength(1) 64. }) 65. it('判断组件的className属性', () => { 66. // props获取所有属性 67. expect(wrapper.props().className) 68. .toBe('App') 69. // console.log(wrapper.props()) 70. }) 71. }) |

## 下午复习

element-react

element-react, element-theme-default

css加载机，字体图标url-loader

create-react-app

create-react-app 项目名称

yarn start

单元测试 jest

describe(intro, fn)

it(intro, fn)

expect(表达式).预测结果

toBe, toEqual, toMatch, toContain, toBeTruthy, toBefalsy ...

生命周期：beforeEach, afterEach, afterAll, beforeAll

enzyme, enzyme-adapter-react-16

configure({ adapter: new Adapter() })

shallow(<App />)

find, contains, props

发布文件 yarn build

输出配置 yarn eject

# 四、项目实战

## 4.1技术架构

react, 路由，axios，jsx， ES6， ES Module， less实现单页面应用程序

## 4.2目录架构

build 项目发布的目录

data 请求mock数据

img 项目所有图片

src 所有开发的文件

app 应用程序

components 所有组件

pages 所有页面

router 路由模块

index.jsx 入口文件

base.less less配置文件

index.html 入口文件

webpack.config.js webpack配置文件

## 4.3路由

有三个页面，对应三个路由

首页是默认路由

详情页：/detail/:id

评论页：/comments/:id

## 4.4 react拓展

每一个组件的属性中，都有一个children属性，表示该组件的内容（子虚拟DOM），我们可以在组件内部渲染它，实现类似vue中slot组件的功能

{this.props.children}

|  |
| --- |
| 1. // 引入库 2. import React, { Component } from 'react'; 3. // 引入路由 4. import { Switch, Route, Redirect } from 'react-router-dom'; 5. // 引入header组件 6. import Header from '../components/header/header' 7. // 引入页面 8. import Home from '../pages/home/home'; 9. import Detail from '../pages/detail/detail'; 10. import Comments from '../pages/comments/comments'; 11. // 默认样式 12. import './app.less'; 13. // 定义组件 14. export default class App extends Component { 15. render() { 16. // 返回虚拟DOM 17. return ( 18. <div> 19. <Header title="爱创课堂新闻平台" rightText="登录" rightClick={e => console.log('success')}></Header> 20. {/\*1 定义路由渲染位置\*/} 21. <Switch> 22. {/\*首页\*/} 23. <Route path="/home" component={Home}></Route> 24. {/\*详情页\*/} 25. <Route path="/detail/:id" component={Detail}></Route> 26. {/\*评论页\*/} 27. <Route path="/comments/:id" component={Comments}></Route> 28. {/\*路由重定向\*/} 29. <Redirect from="\*" to="/home"></Redirect> 30. </Switch> 31. </div> 32. ) 33. } 34. } |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |