**03. 使用webpack3整合antd3的react项目**

[](https://www.jianshu.com/u/86de94c36223)

[码农梦醒](https://www.jianshu.com/u/86de94c36223) 已关注

2018.01.21 23:44\* 字数 297 阅读 1评论 0喜欢 0

**目标**

基于webpack3, antd 3, react 16建立一个react项目, 使用webpack方式整合antd 3, 解决css模块化与antd3的冲突问题. 启用antd3的按需加载

最终目录结构:

|- and-demo-01

|---- app

|---- | ---- index.tmpl.html

|---- | ---- main.js

|---- my\_style

|---- | ---- main.css

|---- node\_modules

|---- public

|---- package.json

|---- webpack.config.js

其中 app为源文件目录, 存放自己编写的js和模板html文件, my\_style存放样式文件. node\_modules是yarn生成的插件目录, public是webpack生成的存放最终编译的css和js的文件夹

**一. 使用yarn创建一个node的package.json文件**

mkdir and-demo-01 && cd and-demo-01

yarn init -y

**二. 引入babel, webpack, less, react插件**

yarn add --dev webpack webpack-dev-server babel-plugin-transform-runtime babel-core babel-loader babel-preset-env babel-preset-react babel-plugin-import style-loader css-loader html-webpack-plugin less-loader less url-loader file-loader

**三. 引入react, antd**

yarn add react react-dom create-react-class antd redux react-redux

**四. 创建webpack.config.js配置文件**

const webpack = require('webpack');

const HtmlWebpackPlugin = require('html-webpack-plugin');

module.exports = {

devtool: 'eval-source-map',

entry: \_\_dirname + "/app/main.js",//已多次提及的唯一入口文件

output: {

path: \_\_dirname + "/public",//打包后的文件存放的地方

filename: "bundle.js"//打包后输出文件的文件名

},

devServer: {

contentBase: "./public",//本地服务器所加载的页面所在的目录

historyApiFallback: true,//不跳转

inline: true, //实时刷新

port: 8080 //设置默认监听端口

},

module: {

rules: [

{

test: /(\.jsx|\.js)$/,

exclude: /node\_modules/,

use: {

loader: "babel-loader",

options: {

presets: [

"env", "react"

],

plugins: [['import', { libraryName: 'antd', style: 'css' }]]

}

}

},

{

test: /\.css$/,

include: /my\_style/, //该目录是自定义的css文件存放目录

use: [

{

loader: "style-loader"

}, {

loader: "css-loader",

options: {

modules: true, // 对自己编写的css文件启用css模块化

localIdentName: '[name]\_\_[local]--[hash:base64:5]' // 指定css的类名格式

}

}

]

},

{

test: /\.css$/,

include: /node\_modules\/antd/,

use: [

{

loader: "style-loader"

}, {

loader: "css-loader",

options: {

modules: false // 因为css模块化会导致antd出现问题,所以对antd的css文件禁用模块化

}

}

]

},

{

test: /\.less$/,

use: [{

loader: 'css-loader'

}, {

loader: 'less-loader' // //因为antd的按需加载使用的less的样式,所以需要less的解析器

}]

},

{

test: /\.jpeg|\.jpg|\.png$/,

exclude: /node\_modules/,

use: [

{

loader: 'url-loader',

options: {

limit: '1024'

}

}

]

}

]

},

plugins: [

new webpack.BannerPlugin('版权所有，翻版必究'),

new HtmlWebpackPlugin({

template: \_\_dirname + "/app/index.tmpl.html"//new 一个这个插件的实例，并传入相关的参数

})

]

}

五. 创建html页面js代码和css代码

app/index.tmpl.html

<!DOCTYPE html>

<html lang="cn">

<head>

<meta charset="utf-8">

<title>首页</title>

</head>

<body>

<div id='root'>

</div>

</body>

</html>

app/main.js

import React, {Component} from 'react';

import ReactDom from 'react-dom';

import createClass from 'create-react-class';

import { Button } from 'antd';

import styles from '../my\_style/main.css';

var Timer = createClass({

getInitialState: function() {

return {

secondsElapsed: Number(this.props.startTime) || 0

};

},

tick: function() { //自定义方法

this.setState({

secondsElapsed: this.state.secondsElapsed + 1

});

},

componentDidMount: function() {//生命周期函数

this.interval = setInterval(this.tick, 1000);

},

componentWillUnmount: function() {//生命周期函数

clearInterval(this.interval);

},

render: function() { //使用JSX返回节点

return (

<div>

Seconds Elapsed: {this.state.secondsElapsed}

</div>

);

}

});

ReactDom.render(<Timer startTime = "60" / >, document.getElementById('root'));

class App extends Component {

render() {

return (

<div className="App">

<Button type="primary" className={styles.redColor}>Button</Button>

</div>

);

}

}

ReactDom.render(<App />, document.getElementById('root'));

my\_style/main.css

html,body{

width:100%;

height:100%;

}

.redColor{

color:red;

font-size: 20px;

}

最终的package.json

{

"name": "react-06",

"version": "1.0.0",

"main": "index.js",

"license": "MIT",

"scripts": {

"start": "webpack",

"server": "webpack-dev-server --open"

},

"devDependencies": {

"babel-core": "^6.26.0",

"babel-loader": "^7.1.2",

"babel-plugin-import": "^1.6.3",

"babel-plugin-transform-runtime": "^6.23.0",

"babel-preset-env": "^1.6.1",

"babel-preset-react": "^6.24.1",

"css-loader": "^0.28.9",

"html-webpack-plugin": "^2.30.1",

"less": "^2.7.3",

"less-loader": "^4.0.5",

"style-loader": "^0.19.1",

"webpack": "^3.10.0",

"webpack-dev-server": "^2.11.1"

},

"dependencies": {

"antd": "^3.1.4",

"create-react-class": "^15.6.2",

"react": "^16.2.0",

"react-dom": "^16.2.0"

}

}

**六. 在and-demo-01执行webpack命令进行编译**

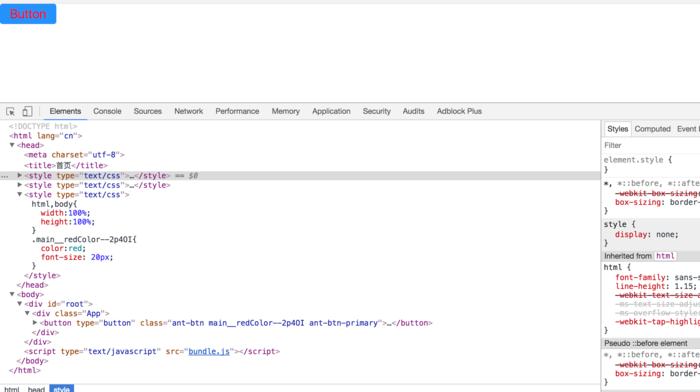
webpack

或

yarn run start

上面的命令会在public目录生成index.html文件和bundle.js文件

双击打开 public/index.html  
最终效果如下:



屏幕快照 2018-01-21 23.40.15.png

按钮显示为蓝色,标识antd按需加载的样式已经生效, 字体为红色标识自定义的css模块化样式也已经生效

以server模式运行(该模式样,server会自动监听js,css,html的变化,自动编辑)

yarn run server

**小礼物走一走，来简书关注我**

https://www.jianshu.com/p/e06044f47ab6