webpack

**一.webpack模块打包器**

**1.目的：**

1.代码转化 es6转化es5

2.文件优化

3.代码分割

4.模块合并

5.自动刷新

6.代码校验

7.自动发布

**2.项目的文件结构**

|-webpack项目名称

|-node\_modules

|-...

|-public

|-admin.html

|-index.html

|-src

|-assets

|-css

|-common

|-public.css

|-index

|-index.css

|-images

|-1.png

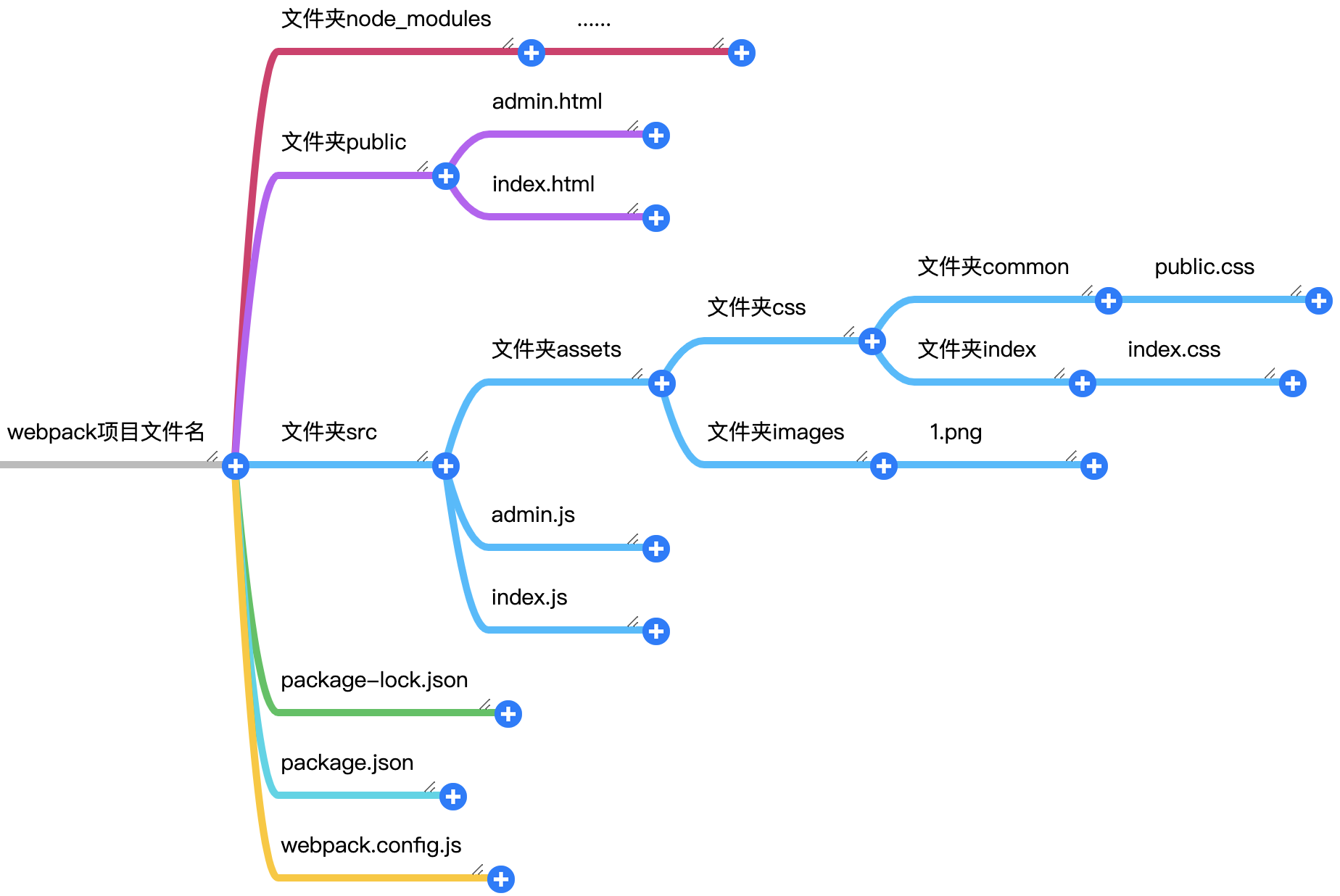
|-admin.js

|-index.js

|-package-lock.json

|-package.json

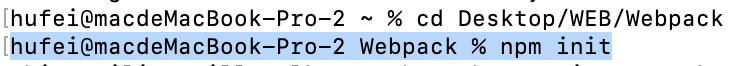
|-webpack.config.js





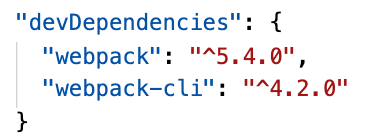
**二.安装webpack基本配置**

**1.进入项目目录生成package.json文件 npm init**



**2.安装webpack和webpack-cli（手脚架）npm install --save-dev webpack@4.32.2 webpack-cli@3.2.2**





**3.执行命令npx webpack 无论err只要能执行了表示安装成功**

**4.配置package.json文件**

"script":{

"build": "webpack --config webpack.config.js" //用于创建上线文件

}



**执行：npm run build 会报错因为没有webpack.config.js文件**

**5.配置入口文件和打包文件**

根目录建立webpack.config.js，创建src文件夹，在src里创建index.js

//webpack是根据node写的所以用node语言

let path = require('path');/ //获取路径

module.exports = {

mode: "development",//这里是开发者环境，build之后是index.js原文显示，如果是production表示生产环境，build会压缩

entry: './src/index.js',//入口文件

output: {

filename: "bundle.js",//打包后的文件名

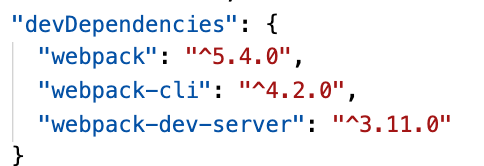
path: path.resolve('dist')//路径设置必须是绝对路径,绝对路径名为dist

}

}

**三.webpack-dev-server的安装与配置服务器**

**1.安装 npm install --save-dev webpack-dev-server**



**2.执行 npx webpack-dev-server**

**3.用package.json配置**

"script":{

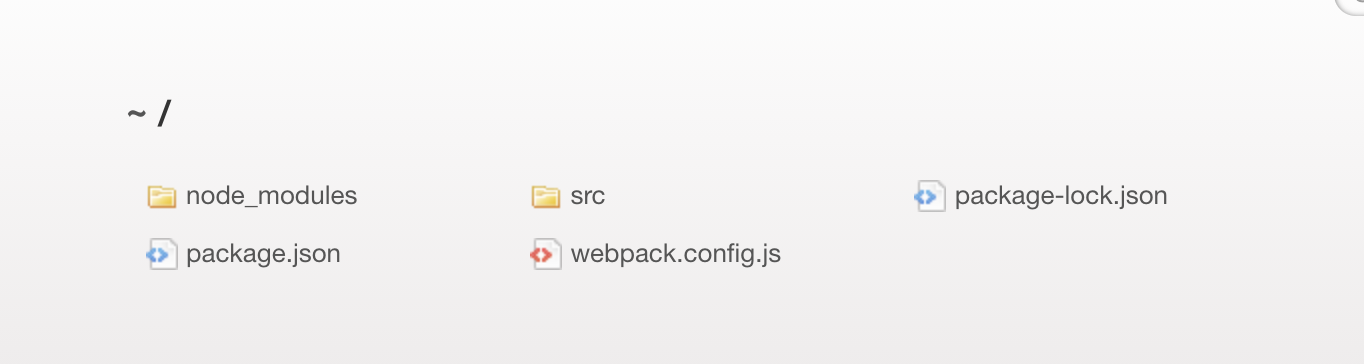
"dev": "webpack-dev-server"

}

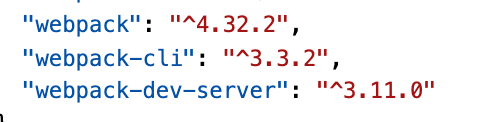


**4.执行：npm run dev**





**注意版本问题卸载重新安装如下版本**



**5.webpack-dev-server的服务器配置（在webpack.config.js文件）**

//webpack是根据node写的所以用node语言

let path = require('path');

// console.log(path.resolve('dist'));

module.exports = {

mode: "production",

entry: './src/index.js',//打包后的文件路口

output: {

filename: "bundle.js",//打包后的文件名

path: path.resolve('dist')//路径设置必须是绝对路径

},

**devServer: {//开启服务器配置**

**port: 8081,//端口**

**host: "localhost",//ip地址：localhost，192.168.10.103可以访问从网络地址**

**progress: true,//开启进度条**

**contentBase: "./dist",//默认打开目录**

**open: true,//自动打开浏览器**

**compress: true//启动gzip压缩**

**}**

}

**6.配置代理解决跨域问题（webpack.config.js）**

let path = require('path');

// console.log(path.resolve('dist'));

module.exports = {

mode: "production",

entry: './src/index.js',//打包后的文件路口

output: {

filename: "bundle.js",//打包后的文件名

path: path.resolve('dist')//路径设置必须是绝对路径

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

**proxy: {**

**'/api': {**

**target: '**http://vueshop.glbuys.com/api**',**

**changeOrigin: true,//是否跨域**

**pathRewrite: {**

**'^/api': ''//重写**

**}**

**}**

**}**

}

}

比如跨域请求http://vueshop.glbuys.com/api/home/index/slide?token=lec949a15fb709370f

在入口文件index.html

跨域请求

$.get("/api/home/index/slide?token=lec949a15fb709370f",function(res){

返回结果res

})

相当于/api取代了http://vueshop.glbuys.com/api

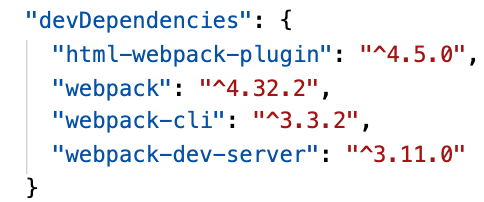
**四.HTML模板插件**

**（单页面SPA）**

使用html模板插件解决启动webpack-dev-server时必须生成build文件夹

1.安装html-webpack-plugin

npm install --save-dev html-webpack-plugin



2.配置html模板插件（先在根目录下创建public文件夹,并创建index文件（就是跨域时的index文件））

（在webpack.config.js）

//webpack是根据node写的所以用node语言

let path = require('path');

// 引入模板插件

**let HtmlWebpackPlugin = require('html-webpack-plugin');**

// console.log(path.resolve('dist'));

module.exports = {

mode: "production",

entry: './src/index.js',//打包后的文件路口

output: {

filename: "bundle.js",//打包后的文件名

path: path.resolve('dist')//路径设置必须是绝对路径

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

**plugins: [**

**new HtmlWebpackPlugin({**

**template: './public/index.html',**

**filename: 'index.html',**

**minify: {**

**collapseWhitespace: true //折叠不换行**

**},**

**hash: true //生产环境的生产hash戳**

**})**

**]**

}

**运行** **npm run dev 就会跳转到index.html页面**

**五.多页面设置（MPA，用的不多,比如一个前台一个后台）**

1.多入口，先在src下家里admin.js,这样就有两个文件还有一个index.js,在public里创建admin.html

webpack.config.js里的entry改成存放多入口的对象

//webpack是根据node写的所以用node语言

let path = require('path');

// 引入模板插件

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

// console.log(path.resolve('dist'));

module.exports = {

mode: "production",

// entry: './src/index.js',//打包后的文件入口

**entry:{**

**index:'./src/index.js',**

**admin:'./src/admin.js'//多入口配置**

**},**

output: {

// filename: "bundle.js",//打包后的文件名

**filename:'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名**

path: path.resolve('dist'),//路径设置必须是绝对路径

**publicPath:'/'//多入口配置，dist之后的公共路径,很重要**

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

**new HtmlWebpackPlugin({**

**template: './public/index.html',**

**filename: 'index.html',**

**chunks:['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题**

**minify: {**

**collapseWhitespace: true**

**},**

**hash: true**

**}),**

**new HtmlWebpackPlugin({**

**template: './public/admin.html',**

**filename: 'admin.html',**

**chunks:['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题**

**minify: {**

**collapseWhitespace: true**

**},**

**hash: true**

**})**

]

}

**六.loaders配置加载css样式**

**1.安装css loader**

npm install --save-dev css-loader@3.2.0 style-loader@1.0.0 mini-css-extract-plugin@0.8.0



css-loader：解析@import这种语法

style-loader：把css插入至head标签中

mini-css-extract-plugin：抽离css样式让index.html里面的css样式变成link引入

**2.配置css loader**

loader是有顺序的默认从右向左，从下往上执行

loader可以写成字符串：use: 'css-loader'，可以写成数组['css-loader']，也可以写成对象[{loader: 'css-loader'}]对象的好处可以传多个参数

**（1）抽离css文件（在src文件夹下创建css文件夹，在css文件夹里创建common文件夹和index文件夹，在common文件夹下创建public.css 在index文件夹下创建index.css）**

let path = require('path');

**//引入抽离css文件**

**let MiniCssExtractPlugin = require('mini-css-extract-plugin');**

// 引入模板插件

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

module.exports = {

mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js',

admin: './src/admin.js'//多入口配置

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/:'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new HtmlWebpackPlugin({

template: './public/admin.html',

filename: 'admin.html',

chunks: ['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

**new MiniCssExtractPlugin({**

**filename: 'static/css/main.css'**

**})**

],

**module: {**

**rules: [**

**{**

**test:/\.css$/,//以.css结尾的文件**

**use:[**

**MiniCssExtractPlugin.loader,//都放到上面的面css里**

**{**

**loader:'css-loader'//最后加载**

**}**

**]**

**}**

**]**

**}**

}

**（2）在入口文件index.js顶部引入index.css （import和require都可以，建议用import）**

require('./assets/css/index/index.css');

**import("./assets/css/index/index.css");**

**（3）在index.css文件顶部引入public.css**

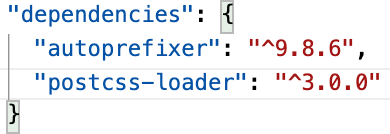
**@import "../common/public.css";**

**七.post-css处理css兼容性（自动加-webkit-）**

css3自动加前缀-webkit-

**1.安装**

npm install --save postcss-loader@3.0.0 autoprefixer@9.8.6



**2.配置**

let path = require('path');

// 抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css3兼容

**let postCss = require('autoprefixer')({**

**"overrideBrowserslist": [**

**'last 10 Chrome versions',**

**'last 5 Firefox versions',**

**'Safari >= 6',**

**'ie > 8'**

**]**

**});**

module.exports = {

mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js',

admin: './src/admin.js'//多入口配置

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/:'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new HtmlWebpackPlugin({

template: './public/admin.html',

filename: 'admin.html',

chunks: ['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

**{**

**loader:'postcss-loader',**

**options:{**

**plugins:[**

**postCss**

**]**

**}**

**}**

]

}

]

}

}

**八.css压缩（删除空格，换行，制表符等）**

1.安装

npm install --save optimize-css-assets-webpack-plugin



2.配置

let path = require('path');

//抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css兼容性

let postCss = require('autoprefixer')({

"overrideBrowserslist": [

'last 10 Chrome versions',

'last 5 Firefox versions',

'Safari >= 6',

'ie > 8'

]

});

**//压缩css**

**let OptimizeCss = require('optimize-css-assets-webpack-plugin');**

module.exports = {

mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js',

admin: './src/admin.js'//多入口配置

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/:'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new HtmlWebpackPlugin({

template: './public/admin.html',

filename: 'admin.html',

chunks: ['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

{

loader: 'postcss-loader',

options: {

plugins: [

postCss

]

}

}

]

}

]

},

**optimization: { // 优化项目动后mode模式代码压缩不再生效，必须同时配置js压缩插件**

**minimizer: [**

**new OptimizeCss() //优化css**

**]**

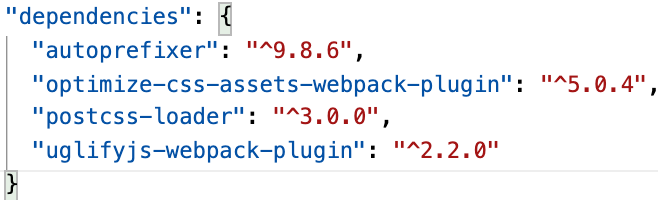
**}**

}

**九.压缩js**

1.下载

npm install --save uglifyjs-webpack-plugin



2.配置

let path = require('path');

//抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css兼容性

let postCss = require('autoprefixer')({

"overrideBrowserslist": [

'last 10 Chrome versions',

'last 5 Firefox versions',

'Safari >= 6',

'ie > 8'

]

});

//压缩css

let OptimizeCss = require('optimize-css-assets-webpack-plugin');

**//压缩js**

**let UglifyjsPlugin = require('uglifyjs-webpack-plugin');**

module.exports = {

// mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js',

admin: './src/admin.js'//多入口配置

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/:'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new HtmlWebpackPlugin({

template: './public/admin.html',

filename: 'admin.html',

chunks: ['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

{

loader: 'postcss-loader',

options: {

plugins: [

postCss

]

}

}

]

}

]

},

optimization: { // 优化项目动后mode模式代码压缩不再生效，必须同时配置js压缩插件

minimizer: [

new OptimizeCss(), //优化css

**new UglifyjsPlugin({**

**cache: true,//是否缓存**

**parallel: true,//是否并发打包**

**sourceMap: true//es6映射es5**

**})**

]

}

}

**十.图片等资源文件处理**

在assets文件夹下创建文件夹images,存入一张图片1.jpg

**1.安装**

npm install --save-dev url-loader



**2.配置**

let path = require('path');

//抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css兼容性

let postCss = require('autoprefixer')({

"overrideBrowserslist": [

'last 10 Chrome versions',

'last 5 Firefox versions',

'Safari >= 6',

'ie > 8'

]

});

//压缩css

let OptimizeCss = require('optimize-css-assets-webpack-plugin');

//压缩js

let UglifyjsPlugin = require('uglifyjs-webpack-plugin');

module.exports = {

// mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js',

admin: './src/admin.js'//多入口配置

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new HtmlWebpackPlugin({

template: './public/admin.html',

filename: 'admin.html',

chunks: ['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

{

loader: 'postcss-loader',

options: {

plugins: [

postCss

]

}

}

]

},

**{**

**test: /\.(png|jpg|jpeg|gif)$/,**

**use: {**

**loader: 'url-loader',//file-loader加载图片，url-loader图片小于多少k用base64显示**

**options: {**

**esModule: false,//防止src变成module对象**

**limit: 100 \* 1024,//小于100k用base64显示**

**outputPath: 'static/images'//build之后的目录分类**

**}**

**}**

**}**

]

},

optimization: { // 优化项目动后mode模式代码压缩不再生效，必须同时配置js压缩插件

minimizer: [

new OptimizeCss(), //优化css

new UglifyjsPlugin({

cache: true,//是否缓存

parallel: true,//是否并发打包

sourceMap: true//es6映射es5

})

]

}

}

**3.使用（在入口文件index.js）**

import("./assets/css/index/index.css");

**var img = new Image(); //这里必须是var因为不支持es6**

**img.src = require("./assets/images/1.jpg");**

**document.body.appendChild(img);**

**十一.babel-loader解决es6相关的问题（转es5）**

**1.安装**

npm install --save babel-loader @babel/core @babel/preset-env @babel/plugin-proposal-class-properties @babel/runtime @babel/plugin-transform-runtime

@babel/core：babel核心文件

@babel/preset-env：es6转es5 只能转低级 let转var

@babel/plugin-proposal-class-properties：支持es6，class Goods类语法

@babel/runtime：编译模板的工具函数

@babel/plugin-transform-runtime：es6转es5时babel会需要一些辅助函数，例如\_extend。这样文件多的时候，项目就会很大。所以babel提供了transform-runtime来将这些辅助函数“搬”到一个单独的模块babel-runtime中，这样做能减小文件大小



**2.配置**

let path = require('path');

//抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css兼容性

let postCss = require('autoprefixer')({

"overrideBrowserslist": [

'last 10 Chrome versions',

'last 5 Firefox versions',

'Safari >= 6',

'ie > 8'

]

});

//压缩css

let OptimizeCss = require('optimize-css-assets-webpack-plugin');

//压缩js

let UglifyjsPlugin = require('uglifyjs-webpack-plugin');

module.exports = {

// mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js',

admin: './src/admin.js'//多入口配置

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new HtmlWebpackPlugin({

template: './public/admin.html',

filename: 'admin.html',

chunks: ['admin'],//admmin.html只引用admin.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

{

loader: 'postcss-loader',

options: {

plugins: [

postCss

]

}

}

]

},

{

test: /\.(png|jpg|jpeg|gif)$/,

use: {

loader: 'url-loader',//file-loader加载图片，url-loader图片小于多少k用base64显示

options: {

esModule: false,//防止src变成module对象

limit: 100 \* 1024,//小于100k用base64显示

outputPath: 'static/images'//build之后的目录分类

}

}

},

**{**

**test:/\.(js|jsx)$/,//支持require js文件（jsx文件）**

**use:{**

**loader:'babel-loader',**

**options:{**

**presets:[**

**'@babel/preset-env'**

**],**

**plugins:[**

**'@babel/plugin-proposal-class-properties',**

**'@babel/plugin-transform-runtime'**

**]**

**}**

**},**

**include:path.resolve(\_\_dirname,'src'), //需要转化的文件夹这里是src文件夹**

**exclude:/node\_modules/ //排除转化的文件**

**}**

]

},

optimization: { // 优化项目动后mode模式代码压缩不再生效，必须同时配置js压缩插件

minimizer: [

new OptimizeCss(), //优化css

new UglifyjsPlugin({

cache: true,//是否缓存

parallel: true,//是否并发打包

sourceMap: true//es6映射es5

})

]

}

}

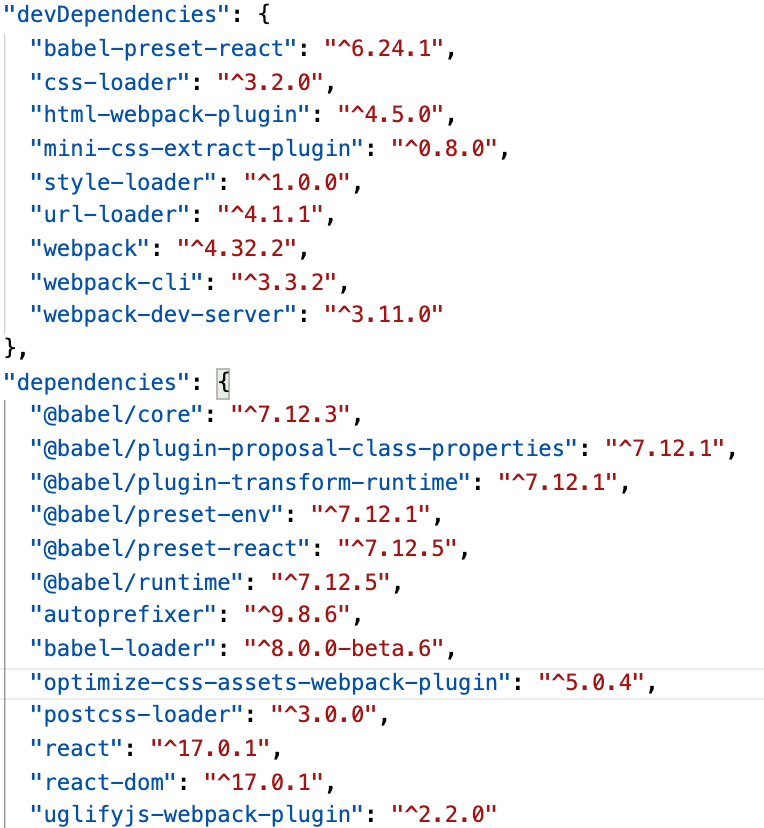
**十二.react项目用webpack配置**

**1.安装所需模块**

react模块：npm i react react-dom --save

react解析模块：npm i babel-preset-react --save-dev

所需的babel：npm install babel-loader@next @babel/core @babel/preset-react @babel/runtime --save



**2.配置(react只支持单页面)**

let path = require('path');

//抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css兼容性

let postCss = require('autoprefixer')({

"overrideBrowserslist": [

'last 10 Chrome versions',

'last 5 Firefox versions',

'Safari >= 6',

'ie > 8'

]

});

//压缩css

let OptimizeCss = require('optimize-css-assets-webpack-plugin');

//压缩js

let UglifyjsPlugin = require('uglifyjs-webpack-plugin');

module.exports = {

// mode: "production",

// entry: './src/index.js',//打包后的文件入口

**entry: {**

**index: './src/index.js'**

**},**

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

{

loader: 'postcss-loader',

options: {

plugins: [

postCss

]

}

}

]

},

{

test: /\.(png|jpg|jpeg|gif)$/,

use: {

loader: 'url-loader',//file-loader加载图片，url-loader图片小于多少k用base64显示

options: {

esModule: false,//防止src变成module对象

limit: 100 \* 1024,//小于100k用base64显示

outputPath: 'static/images'//build之后的目录分类

}

}

},

{

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

use:{

loader:'babel-loader',

options:{

presets:[

'@babel/preset-env',

**'@babel/preset-react'**

],

plugins:[

'@babel/plugin-proposal-class-properties',

'@babel/plugin-transform-runtime'

]

}

},

include:path.resolve(\_\_dirname,'src'), //需要转化的文件夹这里是src文件夹

exclude:/node\_modules/ //排除转化的文件

}

]

},

optimization: { // 优化项目动后mode模式代码压缩不再生效，必须同时配置js压缩插件

minimizer: [

new OptimizeCss(), //优化css

new UglifyjsPlugin({

cache: true,//是否缓存

parallel: true,//是否并发打包

sourceMap: true//es6映射es5

})

]

}

}

**index.html:**



**index.js:**



**十三.webpack脚手架react完整基础配置**

**package.json:**

{

"name": "mywebpack",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"dev": "webpack-dev-server",

"build": "webpack --config webpack.config.js",

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "",

"license": "ISC",

"devDependencies": {

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

"css-loader": "^3.2.0",

"html-webpack-plugin": "^4.5.0",

"mini-css-extract-plugin": "^0.8.0",

"style-loader": "^1.0.0",

"url-loader": "^4.1.1",

"webpack": "^4.32.2",

"webpack-cli": "^3.3.2",

"webpack-dev-server": "^3.11.0"

},

"dependencies": {

"@babel/core": "^7.12.3",

"@babel/plugin-proposal-class-properties": "^7.12.1",

"@babel/plugin-transform-runtime": "^7.12.1",

"@babel/preset-env": "^7.12.1",

"@babel/preset-react": "^7.12.5",

"@babel/runtime": "^7.12.5",

"autoprefixer": "^9.8.6",

"babel-loader": "^8.0.0-beta.6",

"optimize-css-assets-webpack-plugin": "^5.0.4",

"postcss-loader": "^3.0.0",

"react": "^17.0.1",

"react-dom": "^17.0.1",

"uglifyjs-webpack-plugin": "^2.2.0"

}

}

**webpack.config.js:**

let path = require('path');

//抽离css文件

let MiniCssExtractPlugin = require('mini-css-extract-plugin');

// 引入模板插件

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

// 处理css兼容性

let postCss = require('autoprefixer')({

"overrideBrowserslist": [

'last 10 Chrome versions',

'last 5 Firefox versions',

'Safari >= 6',

'ie > 8'

]

});

//压缩css

let OptimizeCss = require('optimize-css-assets-webpack-plugin');

//压缩js

let UglifyjsPlugin = require('uglifyjs-webpack-plugin');

module.exports = {

// mode: "production",

// entry: './src/index.js',//打包后的文件入口

entry: {

index: './src/index.js'

},

output: {

// filename: "bundle.js",//打包后的文件名

filename: 'static/js/[name].js',//多入口配置，name会自动匹配entry里的键名

path: path.resolve('dist'),//路径设置必须是绝对路径

publicPath: '/'//多入口配置，dist之后的公共路径,很重要

},

devServer: {//开启服务器配置

port: 8081,//端口

host: "localhost",//ip地址：localhost，0.0.0.0可以访问网络地址

progress: true,//开启进度条

contentBase: "./dist",//默认打开目录

open: true,//自动打开浏览器

compress: true,//启动gzip压缩

//跨域

proxy: {

'/api': {

target: 'http://vueshop.glbuys.com/api',//跨域的地址

changeOrigin: true,//是否跨域

pathRewrite: {

'^/api': ''//重写

}

}

}

},

plugins: [

new HtmlWebpackPlugin({

template: './public/index.html',

filename: 'index.html',

chunks: ['index'],//index.html只引用index.js,解决index.html里有index.js和admin.js的问题

minify: {

collapseWhitespace: true

},

hash: true

}),

new MiniCssExtractPlugin({

filename: 'static/css/main.css'

})

],

module: {

rules: [

{

test: /\.css$/,//以.css结尾的文件

use: [

MiniCssExtractPlugin.loader,//都放到上面的面css里

{

loader: 'css-loader'//最后加载

},

{

loader: 'postcss-loader',

options: {

plugins: [

postCss

]

}

}

]

},

{

test: /\.(png|jpg|jpeg|gif)$/,

use: {

loader: 'url-loader',//file-loader加载图片，url-loader图片小于多少k用base64显示

options: {

esModule: false,//防止src变成module对象

limit: 100 \* 1024,//小于100k用base64显示

outputPath: 'static/images'//build之后的目录分类

}

}

},

{

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

use:{

loader:'babel-loader',

options:{

presets:[

'@babel/preset-env',

'@babel/preset-react'

],

plugins:[

'@babel/plugin-proposal-class-properties',

'@babel/plugin-transform-runtime'

]

}

},

include:path.resolve(\_\_dirname,'src'), //需要转化的文件夹这里是src文件夹

exclude:/node\_modules/ //排除转化的文件

}

]

},

optimization: { // 优化项目动后mode模式代码压缩不再生效，必须同时配置js压缩插件

minimizer: [

new OptimizeCss(), //优化css

new UglifyjsPlugin({

cache: true,//是否缓存

parallel: true,//是否并发打包

sourceMap: true//es6映射es5

})

]

}

}