const ManifestPlugin = require('webpack-manifest-plugin'); //生成manifest.json;

const webpack = require('webpack');

const UglifyJSPlugin = require('uglifyjs-webpack-plugin');

const config = require('../config/config');

const path = require('path');

const CopyWebpackPlugin = require('copy-webpack-plugin');

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

console.log('压缩')

const ExtractTextPlugin = require("extract-text-webpack-plugin"); //Extract text from a bundle, or bundles, into a separate file.

// 为每个页面定义一个 ExtractTextPlugin

const indexExtractCss = new ExtractTextPlugin('index/[name].[contenthash].css')

const detailExtractCss = new ExtractTextPlugin('detail/[name].[contenthash].css')

module.exports = {

entry: {

index: path.resolve(\_\_dirname, '../src/index/js/index.js'),

detail: path.resolve(\_\_dirname, '../src/detail/js/detail.js')

},

output: {

filename: assetsPath("js/[name].[hash].js"),

path: path.resolve(\_\_dirname, config.build.assetsRoot),

publicPath: '../'

},

module: {

rules: [

// 每个页面的 ExtractTextPlugin 只处理这个页面的样式文件

{

test: /src(\\|\/)index(\\|\/)css(\\|\/).\*\.(css|scss)$/,

use: indexExtractCss.extract({

fallback: 'style-loader',

use: ['css-loader']

})

}, {

test: /src(\\|\/)detail(\\|\/)css(\\|\/).\*\.(css|scss)$/,

use: detailExtractCss.extract({

fallback: 'style-loader',

use: ['css-loader']

})

}, {

test: /\.(cur|ico|gif|jpe?g|png|svg)(\?.\*)?$/,

loader: "url-loader",

options: {

limit: 500000,

name: "images/[name].[hash:7].[ext]",

}

},

{

test: /\.html$/,

use: [{

loader: 'html-loader',

// options: {

// minimize: true

// }

}],

}

]

},

plugins: [

//针对每个入口项目，单独引入css

indexExtractCss,

detailExtractCss,

//单独处理每一个入口的html

new HtmlWebpackPlugin({

chunks:['index'], //添加引入的js,也就是entry中的key ，这一句是关键，没有的话会把entry中的文件都打包进来

filename: 'index/index.html',

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

inject: true,

minify: {

removeComments: true,

collapseWhitespace: false

}

}),

new HtmlWebpackPlugin({

chunks:['detail'],

filename: 'detail/detail.html',

template: 'src/detail/html/detail.html',

inject: true,

minify: {

removeComments: true,

collapseWhitespace: false

}

}),

new ManifestPlugin(),

new UglifyJSPlugin(),

// new webpack.optimize.CommonsChunkPlugin({

// name: "vendor",

// minChunks: function (module, count) {

// // any required modules inside node\_modules are extracted to vendor

// return module.resource &&

// /\.js$/.test(module.resource) && (

// module.resource.indexOf(path.join("/node\_modules/")) != -1 ||

// module.resource.indexOf(path.join("/shared/")) != -1

// );

// }

// }),

// extract webpack runtime and module manifest to its own file in order to

// prevent vendor hash from being updated whenever app bundle is updated

// new webpack.optimize.CommonsChunkPlugin({

// name: "manifest",

// chunks: ["vendor"]

// }),

//拷贝第三方库到dist目录

new CopyWebpackPlugin([{

from: path.resolve(\_\_dirname, '../third-lib'),

to: path.resolve(\_\_dirname, config.build.assetsRoot, 'third-lib'),

ignore: ['.\*']

}])

]

}

function assetsPath(\_path) {

var assetsSubDirectory = config.build.assetsSubDirectory

return path.posix.join(assetsSubDirectory, \_path)

}