***Vue***

**webpack**

webpack是一个模块打包器。

全局安装webpack，才能使用webpack的命令

全局安装webpack-cli

新建webpack的配置文件，默认是webpack.config.js

webpack 基于nodejs 的。默认只识别js文件。

**四个核心概念:**

**1.入口(entry)**

entry：string | array | object

前两种是适用于一个入口文件，最后一个是适用于多入口文件

entry:{

key1:value1, //value指的是入口路径

key2:value2

}

**2.出口(output)**

默认的出口文件夹是dist，编译之后的文件名是main.js。

output:{

path:path.join(\_\_dirname,"build"), //编译之后的出口路径

filename:'编译之后的文件名字' // filename:”[name]-[hash].js”//多入口

}

**3.加载器(loader)**

module:{

rules:[

{test:/\.(js | jsx)$/}, //匹配文件,需要注意test后边不需要加引号

loader:’babel-loader’,

options:{

presets:[‘@babel/preset-env’]

},

use:[‘style-loader’,’css-loader’]， //执行顺序是从右往左的

options: {

name: "./font/[name].[ext]"

}

}，

{},

......

]

}

.babelrc文件 => {

“presets”:[

“@babel/preset-env”

]

}

**4.插件(plugins)**

* 下载插件
* 引入插件
* 使用插件
* 调用插件

extract-text-webpack-plugin 将捆绑包或捆绑包中的文本提取到单独的文件中。

plugins: [

new ExtractTextPlugin("style.css"),

]

clean-webpack-plugin 用于删除/清除构建文件夹的webpack插件。

plugins: [

new CleanWebpackPlugin(),

]

html-webpack-plugin 生成一个 HTML5 文件， 其中包括使用 script 标签的 body 中的所有 webpack 包。

plugins: [

new ExtractTextPlugin("style.css"),

new CleanWebpackPlugin(),

new HtmlWebpackPlugin({

template: "index.html", //模板

filename: "app.html", //构建之后的文件名

title: "首页",

minify: {

removeAttributeQuotes: true, //去除引号

collapseWhitespace: true, //去除空格

removeComments: true, //去除注释

removeEmptyAttributes: true, //去除空属性

}

})

]

**抽离css**

1.下载插件extract-text-webpack-plugin

下载的时候需注意版本问题 extract-text-webpack-plugin@next

2.使用插件

在module里面

Use:ExtractTextPlugin.extract({

fallback: "style-loader", //运行结束时加载的loader

use: ["css-loader","sass-loader"] //若是抽离单纯的css，直接css-loader

})

3.调用

在plugins: [

new ExtractTextPlugin("style.css") //抽离之后的文件名称

]

webpack --config webpack.dev.js //当配置文件名不是默认的时候执行

mode:’development’, //设置webpack环境 ‘development’开发环境 ‘production’生产环境

package.json配置命令：

“scripts”:{

“build”:”webpack --config webpack.dev.js --mode development” //“webpack --mode=development”

}

npm run build 执行命令

**补：模块**

抛出 export || export default

导入 import 名字 from 路径

as 关键字 重命名

一个js文件里面，只能有一个默认抛出

默认抛出的，在接收的时候，要放在第一个参数接收

**配置**

webpack-dev-server（简写为：dev-server） 能够用于快速开发应用程序。

1.下载

npm i webpack-dev-server -D || npm i webpack-dev-server -g

2.执行

webpack-dev-server --config webpack.dev.js

在package.json中配置scripts：

"scripts": {

"dev": "webpack-dev-server --config webpack.dev.js"

},

需注意webpack-dev-server要和webpack一起使用，即两者要一起在本地或者全局。

3.参数

devServer: { //开发环境

port: 9001,

open: true,

host:"localhost",

inline:true,

before(app){ //与middleware作用相似

app.get('/api/list',(req,res,next) => {

res.send({code:2,msh:"成功"})

})

}，

proxy:{ //代理，类似于proxies

“/classify/iconlist”:”http://loaclhost:3000”

}

}

**模块热替换插件（HotModuleReplacementPlugin）**

HotModuleReplacementPlugin属于webpack的内置插件，所以不需要下载

1.引入webpack

const webpack = require(“webpack”)

2.插件

plugins:[

new webpack.HotModuleReplacementPlugin()

],

devServe:{

hot:true //同inline:true作用相同，二者只写其一

}

**环境分离**

**目录**

build文件夹

webpack.base.js 公共部分

webpack.dev.js 开发环境

webpack.product.js 生产环境

config文件夹 整个项目的配置信息

index.js

**devtool**

开发过程中才能打断点，只能在原始代码（没有压缩之前）中打断点

development:{

devTool:’cheap-module-eval-source-map’

},

production:{

devTool:’source-map’

}