[vue2+3+webpack]

[2021.12.22]

文档标题作为文字占位，只需单击占位更换文本添加内容

摘要

[前端框架]

[日期及邮编地址]

[ID名称]

[vue2 3](#_Toc16854)

[webpack 3](#_Toc27344)

[初始化项目 3](#_Toc5467)

[在项目中安装webpack 3](#_Toc28800)

[dependencies和devDependencies节点 3](#_Toc11572)

[在项目中配置webpack 3](#_Toc12643)

[webpack中的默认约定（即webpack.config.js文件中的初始默认配置） 4](#_Toc18403)

[自定义打包的入口与出口 4](#_Toc25964)

[webpack插件 4](#_Toc19236)

[webpack-dev-server 4](#_Toc31412)

[安装webpack-dev-server 4](#_Toc14031)

[配置webpack-dev-server 4](#_Toc6983)

[webpack-dev-server生成的文件位置 4](#_Toc30705)

[html-webpack-plugin 4](#_Toc1648)

[安装html-webpakc-plugin 5](#_Toc319)

[配置html-webpakc-plugin 5](#_Toc15072)

vue2

webpack

webpack是前端工程化的具体解决方案

他提供了友好的前端模块化开发支持，以及代码压缩混淆、处理浏览器端JavaScript的兼容性、性能优化等强大功能

需要安装nodejs才能使用

初始化项目



注：-S表示将安装的包名和版本号记录在package.json文件的dependencies节点下，不写的 话也会记录。-S是--save的简写

install可以简写为i

在项目中安装webpack

npm install webpack@5.42.1 webpack-cli@4.7.2 -D

注：-D表示将安装的包名和版本号记录到devDependencies节点下

-D是--save-dev的简写

@用来指定版本号

dependencies和devDependencies节点

dependencies：开发和生产环境用到的包记录在该目录下

devDependencies：只有开发阶段会被用到的包记录在该目录下

在项目中配置webpack

1. 在项目根目录创建名为webpack.config.js的webpack配置文件，并初始化如下的基本配置

module.exports = {

mode: ’development’, //mode用来指定构建模式，有development和production

}

该配置控制webpack做不做代码压缩，development不压缩，production压缩

1. 在package.json的scripts节点下，新增dev脚本

“scripts”: {

“dev”: “webpack”, //scripts节点下的脚本可以通过npm run执行，如npm run dev

}

1. 在终端中运行npm run dev命令，启动webpack进行项目的打包构建

打包完成后会生成dist文件夹，里面有生成的main.js文件，该文件就是webpack打包的综合性js文件，包含用户自己写的代码和用到的工具包代码，可以解决低版本浏览器不兼容高级代码的问题

webpack中的默认约定（即webpack.config.js文件中的初始默认配置）

在webpack4.x 和5.x中

1. 默认打包的入口文件为src/index.js
2. 默认的输出文件路径为dist/main.js

这两种配置可以在webpack.config.js内修改

自定义打包的入口与出口

通过配置webpack.config.js文件中的entry节点指定打包的入口，通过output节点指定打包的出口，示例代码如下

const path = require(‘path’); //导入nodejs中专门操作路径的模块

module.exports = {

entry: path.join(\_\_dirname, ‘./src/index.js’);

output: {

path: path.join(\_\_dirname, ‘./dist’), //输出文件的存放路径

filename: ‘main.js’, //输出的文件名

webpack插件

通过安装和排至第三方的插件，可以拓展webpack的能力，从而让webpack用起来更方便。最常用的webpack插件有两个：

webpack-dev-server

类似于nodejs中的nodemon工具

每当修改了源代码，webpack会自动进行项目的打包和构建

安装webpack-dev-server

npm install webpack-dev-server@3.11.2 -D

配置webpack-dev-server

1. 修改package.json文件中scripts的dev命令如下

“scripts” : {

“dev”: “webpack serve”,

}

1. 再次运行npm run dev重新进行项目的打包
2. 浏览器中访问localhost:8080可以查看打包效果

webpack-dev-server生成的文件位置

webpack-dev-server生成的文件不会直接保存在硬盘上（照顾硬盘的读写寿命），所以在资源管理器上看不到它，生成的文件被保存在内存的根目录下，即/，引用路径直接写/xx.js即可

html-webpack-plugin

webpack中的html插件（类似于一个模板引擎插件），配合webpack-dev-server使用

可以通过此插件自定制index.html页面的内容，本质是把src内的index.html复制到根目录下，如此一来访问localhost:8080时会直接加载index.html

安装html-webpakc-plugin

npm install html-webpakc-plugin@5.3.2 -D

配置html-webpakc-plugin

在webpack.config.js文件内配置

//导入插件，得到一个构造函数

const HtmlPlugin = require(‘html-webpack-plugin’)

//创建插件实例对象

const htmlPlugin = new HtmlPlugin({

template: ‘./src/index.html’, //指定原文件的存放路径

filename: ‘./index.html’, //指定生成的文件的存放路径

})

module.exports = {

mode: ‘development’,

plugins: [htmlPlugin], //通过plugins节点，使htmlPlugin插件生效

}