



yanyongchao

2018 年 03 月 25 日

webpack4之基础篇

1. 什么是webpack

WebPack可以看做是模块打包机：它做的事情是，分析你的项目结构，找到JavaScript模块以及其它的一些浏览器不能直接运行的拓展语言（Scss，TypeScript等），并将其打包为合适的格式以供浏览器使用。

2. webpack核心概念

1. Entry：入口，Webpack 执行构建的第一步将从 Entry 开始，可抽象成输入。
2. Module：模块，在 Webpack 里一切皆模块，一个模块对应着一个文件。Webpack 会从配置的 Entry 开始递归找出所有依赖的模块。
3. Chunk：代码块，一个 Chunk 由多个模块组合而成，用于代码合并与分割。Loader：模块转换器，用于把模块原内容按照需求转换成新内容。
4. Plugin：扩展插件，在 Webpack 构建流程中的特定时机注入扩展逻辑来改变构建结果或做你想要的事情。
5. Output：输出结果，在 Webpack 经过一系列处理并得出最终想要的代码后输出结果。

3. webpack执行流程

webpack启动后会在entry里配置的module开始递归解析entry所依赖的所有module，没找到一个module，就会根据配置的loader去找相应的转换规则，对module进行转换后在解析当前module所依赖的module，这些模块会以entry为分组，一个entry和所有相依赖的module也就是一个chunk，最后webpack会把所有chunk转换成文件输出，在整个流程中webpack会在恰当的时机执行plugin的逻辑

4. 开始从零配置webpack



```
npm init
```

2. 配置webpack

```
yarn add webpack webpack-cli -D //webpack4把webpack拆分了
mkdir src    cd src并且创建index.html, index.js
mkdir dist
touch webpack.config.js
```

webpack.config.js

```
module.exports = {
  entry: './src/index.js', //入口文件, src下的index.js
  output: {
    path: path.join(__dirname, 'dist'), // 出口目录, dist文件
    filename: '[name].[hash].js' //这里name就是打包出来的文件名, 因为是单入口, 就是main, 多入口下回分
  },
  module: {},
  plugin: {},
  devServer: {}
}
```

3. 配置开发服务器

```
yarn add webpack-dev-server -D
修改webpack.config.js
devServer: {
  contentBase: path.join(__dirname, "dist"), //静态文件根目录
  port: 9090, // 端口
  host: 'localhost',
  overlay: true,
  compress: true // 服务器返回浏览器的时候是否启动gzip压缩
}
修改package.json
"script": {
  "build": "webpack --mode development", //这里为了不压缩代码, 用开发环境
  "dev": "webpack-dev-server --open --mode development"
}
```

4. 支持css文件



```
// style-loader可以把css文件变成style标签插入head中
// 多个loader是有顺序要求的，从右往左写，因为转换的时候是从右往左转换的
module: {
  rules: {
    test: /\.css$/,
    use: ['style-loader', 'css-loader'],
    include: path.join(__dirname, 'src'), //限制范围，提高打包速度
    exclude: /node_modules/
  }
}
```

5. 支持es6, react.js等

```
yarn add babel-loader babel-core babel-preset-env babel-preset-stage-0 babel-preset -react -D
{
  test: /\.js$/,
  use: {
    loader: 'babel-loader',
    query: {
      presets: ['env', 'stage-0', 'react'] // env转换es6 stage-0转es7 react转react
    }
  }
}
同时可以把babel配置写到.babelrc中
```

6. 从js中分离css

```
yarn add extract-text-webpack-plugin -D
{
  test: /\.css$/, // 转换文件的匹配正则
  use: ExtractTextWebpackPlugin.extract({
    fallback: 'style-loader',
    //如果需要，可以在 sass-loader 之前将 resolve-url-loader 链接进来
    use: ['css-loader']
  })
},
//加上plugin
plugins: [
  new ExtractTextWebpackPlugin({
    filename: 'css/[name].[hash].css' //放到dist/css/下
  })
]
```



`url-loader` 当图片较小的时候会把图片BASE64编码，大于`limit`参数的时候还是使用`file-loader` 进行拷贝

```
{
  // file-loader是解析图片地址，把图片从源文件拷贝到目标文件并且修改源文件名字
  // 可以处理任意二进制，bootstrap里的字体
  // url-loader可以在文件比较小的时候，直接变成base64字符串内嵌到页面中
  {
    test: /\.?(png|jpg|jpeg|gif|svg)/,
    use: {
      loader: 'url-loader',
      options: {
        outputPath: 'images/', // 图片输出的路径
        limit: 5 * 1024
      }
    }
  },
  // 同时要处理打包图片路径问题，
  output: {
    publicPath: '/'
  }
}
```

8. 处理css3属性前缀

yarn add postcss-loader -D

```
{
  test: /\.css$/, // 转换文件的匹配正则
  // css-loader用来处理css中url的路径
  // style-loader可以把css文件变成style标签插入head中
  // 多个loader是有顺序要求的，从右往左写，因为转换的时候是从右往左转换的
  // 此插件先用css-loader处理一下css文件
  use: ExtractTextWebpackPlugin.extract({
    fallback: 'style-loader',
    //如果需要，可以在 sass-loader 之前将 resolve-url-loader 链接进来
    use: ['css-loader', 'postcss-loader']
  })
},
建立.postcssrc.js文件
module.exports = {
  "plugins": {
    // to edit target browsers: use "browserslist" field in package.json
    "autoprefixer": {
      "browsers": [
        "ie >= 9",
        "ff >= 30",
        "chrome >= 34",

```



```
}
```

9. 调试打包的代码 webpack通过配置可以自动给我们source maps文件，map文件是一种对应编译文件和源文件的方法

devtool: 'eval-source-map'

1. `source-map` 把映射文件生成到单独的文件，最完整最慢
2. `cheap-module-source-map` 在一个单独的文件中产生一个不带列映射的Map
3. `eval-source-map` 使用`eval`打包源文件模块，在同一个文件中生成完整sourcemap
4. `cheap-module-eval-source-map` sourcemap和打包后的JS同行显示，没有映射列

10. 压缩js

webpack --mode production 会压缩，可以忽略下面

```
yarn add uglifyjs-webpack-plugin -D
```

```
const UglifyjsWebpackPlugin = require('uglifyjs-webpack-plugin')  
new UglifyjsWebpackPlugin(),
```

11. 压缩css

webpack --mode production 会压缩，可以忽略下面

```
{  
  test: /\.css$/, // 转换文件的匹配正则  
  use: ExtractTextWebpackPlugin.extract({  
    fallback: 'style-loader',  
    use: [  
      {  
        loader: 'css-loader',  
        options: {minimize: true}  
      },  
      'postcss-loader'  
    ]  
  })  
},
```

12. 清空打包输出目录

```
yarn add clean-webpack-plugin -D
```

```
const CleanWebpackPlugin = require('clean-webpack-plugin')
```



量：

```
"scripts": {
  "build": "cross-env NODE_ENV=production webpack --mode development", // 设置NODE_ENV为production
  "dev": "webpack-dev-server --open --mode development "
},
new webpack.DefinePlugin({
  NODE_ENV: JSON.stringify(process.env.NODE_ENV)
})
```

在全局都有NODE_ENV这个变量,当yarn run build,时, NODE_ENV = 'production'

14. 暴露全局变量

```
new Webpack.ProvidePlugin({
  '$': 'jquery'
}),
```

15. resolve解析

extension: 指定extension之后可以不用在require或是import的时候加文件扩展名,会依次尝试添加扩展名进行匹配

```
resolve: {
  //自动补全后缀,注意第一个必须是空字符串,后缀一定以点开头
  extension: ['', '.js', '.json', '.css']
}
```

alias: 配置别名可以加快webpack查找模块的速度

```
resolve: {
  alias: {
    'bootstrap': 'bootstrap/dist/css/bootstrap.css'
  }
}
```

16. 复制静态资源

```
yarn add copy-webpack-plugin
const CopyWebpackPlugin = require('copy-webpack-plugin')
new CopyWebpackPlugin([
  {
    from: path.resolve(__dirname, 'static'),
    to: path.resolve(__dirname, 'pages/static'),
    ignore: ['.*']
  }
])
```



关注下面的标签，发现更多相似文章

[JavaScript](#)[Webpack](#)

[掘金招聘前端开发、后端开发、运营经理、内容运营](#)

加入掘金和开发者一起成长。发送简历到 hr@xitu.io，期待你的加入！

评论

说说你的看法

dissswrod

这是我见过的总结最好的webpack配置

▲ 0 评论 刚刚

badflockmaster 前端 @ 读书中

厉害厉害

▲ 0 评论 1月前

沙正虎 前端 @

zan

▲ 0 评论 1月前

Kalecgos

为作者疯狂打call

▲ 1 评论 1月前

godzbin

疯狂打call

[首页](#) ▾

相关推荐

专栏 · nw2018 · 2天前 · Node.js / 前端

从零开始搭建一个mock服务

♥ 14 💬 3

专栏 · Hollis_公众号Hollis · 1小时前 · 后端 / 服务器

大家都在说的分布式系统到底是什么？

♥ 13 💬

专栏 · 棕小渐 · 2天前 · JavaScript / 前端

函数式编程（JavaScript描述）

♥ 9 💬

专栏 · MarsDes · 2天前 · 微信小程序 / 前端

小程序 音频API采坑完全手册

♥ 6 💬

热 · 专栏 · 热心市民老贾 · 18小时前 · JavaScript

一个Promise面试题

♥ 167 💬 11

热 · 专栏 · 老姚 · 2天前 · CSS

chrome开发者工具各种骚技巧

♥ 1188 💬 57

专栏 · wangpf · 2天前 · JavaScript / 前端

由Object.prototype.toString.call()引发关于toString()方法的思考

♥ 1 💬

热 · 专栏 · xietao3 · 1天前 · React.js

一份传男也传女的 React Native 学习笔记

♥ 170 💬 21



一个帮助开发者成长的社区





▼ 445

▼ 13

热 · 专栏 · 龙逸轩 · 5天前 · Vue.js

VUE组件汇总

 1052

 23

