Webpack 技术简要

2021年11月1日 星期一 下午 4:13

1. basic：

entry output 入口 输出 默认打包js

2.loder:

css less-loader

图片的 file-loader url-loader(can 64bit)

图片loader集成的asset模块

font文件woff打包 --可以file-loader也可以asset/resource

3.plugins 找插件->实现功能

clean-webpack-plugin删除build再更新

html-webpack-plugin 打包/自动生成index.html

DefinePlugin 配置全局常量，给html模板编写提供支持

copy-webpack-plugin 文件夹复制，可设置忽略

4. webpack其他配置

mode:"development",

devtool: "source-map" 默认eval

5. babel

babel：本质是compiler，（看抽象语法树）

webpack是要和babel一起使用的，只不过脚手架cli封装好了。作用：旧代码向后兼容、语法转换等。

需要转换什么语法就需要什么插件，要npm安装

ppt里，github有个优秀的compiler lispToC，有时间去看看。

使用： webpackConfig: babel-loader option:preset ...

也可以单独写一个babel.config.json

6.vue模块

import {createApp } from 'vue'(npm install vue@next)

不用cdn src导入vue文档，通过包导入

vue打包后的版本：

vue[.runtime].global[.prod] (可通cdn src，全局vue)

prod:压缩 runtime：不包含compiler，更小

vue[.runtime].esm-browser[.prod] (type module)

esm-browser: 原生ES模块导入

vue.runtime.esm-bundler.js (webpack等)

构建工具默认，如果需要.html .js 中用template模块组件，要手动指定vue.esm-bundler.js ，组件写在.vue文件中则不需，直接vue导入

import { createApp } from 'vue/dist/vue.esm-bundler';

export default: import +任意名字 = import 都是导入default这个事务

webpack配置Vue： @vue/comiler-sfc (vueLoader支持库) + vue-loader@next + 导入VueLoaderPlugin from vue-loader/dist/index + new VueLoaderPlugin()使用

.vue style 默认关键字scoped

template等是option from Vue2， 默认是支持的，但是强烈建议自己配置一下， 在插件plugin里的DefinePlugin 设置，

```
new DefinePlugin({
    BASE_URL: "'./'",
    __VUE_OPTIONS_API__: true,
    __VUE_PROD_DEVTOOLS__: false （生产阶段不需要devtools）
  }),
```

7. devServer

希望webpack监听我的代码——搭建本地服务，替换热模块

① watch mode

在package.json中， scripts： webpack --watch

Or 在webpackConfig中加入watch：true

实际是通过vscode插件的live-server来完成的自动刷新浏览器

②webpack-dev-server(常用) live reloading

安装 Webpack server

然后 package.json 改scripts，达成npm run serve

dev打包到内存，供chrome直接访问，并不输出文件

contentBase

配置devServer

target : "web"; //为 什么环境打包的

```
devServer: {
    contentBase: "./public", //serve 在src找不到就去public找，不然就需要CopyWebpackPlugin插件来帮忙
    hot: true, // hot module replacement 热模块替换 提高开发效率，不开的话就会修改即刷新页面
//热替换： module.hot.accept("",()); 指定模块热替换  但是Vue和React的loader已经有HMR，不需要额外指定
    host: "0.0.0.0",//默认是 'localhost' ，127.0.0.1, 回环地址，别的主机无法访问。0.0.0.0是ipv4全部地址，外面可访问
    port: 7777, //端口
    open: true, //webpack serve --open也可以 是否打开浏览器
    // compress: true, 是否压缩，gzip压缩，浏览器默认支持，传输更快
    proxy: {
      "/api": {
        target: "http://localhost:8888", //代理地址，跨域访问，代理server访问server
        pathRewrite: {
          "^/api": "" //路径重写  http://localhost:7777(/api)/moment
        },
```

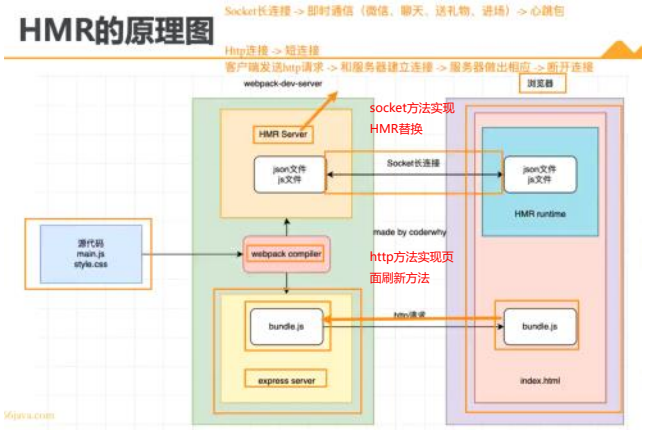
```
//插件导入
const path = require("path");
const { CleanWebpackPlugin } = require("clean-webpack-plugin");
const HtmlWebpackPlugin = require("html-webpack-plugin");
const { DefinePlugin } = require("webpack");
const CopyWebpackPlugin = require('copy-webpack-plugin');

module.exports = {
  // 设置模式
  // development 开发阶段，会设置development
  // production 准备打包上线的时候，设置production
  mode: "development",
  // 设置source-map，建立js映射文件，方便调试代码和错误
  devtool: "source-map",

  entry: "./src/main.js",
  output: {
    path: path.resolve(__dirname, "./build"),
    filename: "js/bundle.js",
    // assetModuleFilename: "img/[name]_[hash:6][ext]"
  },
  module: {// 设置文件处理规则
    rules: [
      {
        test: /\.css$/,
        use: ["style-loader", "css-loader", "postcss-loader"],
      },
      {
        test: /\.less$/,
        use: ["style-loader", "css-loader", "less-loader"],
      },
      // },
      {
        test: /\.(jpe?g|png|gif|svg)$/,
        type: "asset",
        generator: {
          filename: "img/[name]_[hash:6][ext]",
        },
        parser: {
          dataUrlCondition: {
            maxSize: 10 * 1024,
          },
        },
      },
      {
        test: /\.(eot|ttf|woff2?)$/,
        type: "asset/resource",
        generator: {
          filename: "font/[name]_[hash:6][ext]",
        },
      },
      // {
      //   test: /\.js$/,
      //   use: {
      //     loader: "babel-loader",
      //     options: {
      //       // plugins: [
      //       //   "@babel/plugin-transform-arrow-functions",
      //       //   "@babel/plugin-transform-block-scoping",
      //       // ]
      //       presets: [
      //         "@babel/preset-env"
      //       ]
      //     }
      //   }
      // }
      {
        test: /\.js$/,
        loader: "babel-loader"
      }
    ],
  },
  plugins: [
    new CleanWebpackPlugin(),
    new HtmlWebpackPlugin({
      template: "./public/index.html",
      title: "哈哈哈哈"
    }),
    new DefinePlugin({
      BASE_URL: "'./'"
    }),
    new CopyWebpackPlugin({
      patterns: [
        {
          from: "public",
          to: "./",
          globOptions: {
            ignore: [
              "**/index.html"
            ]
          }
        }
      ]
    })
  ],
};
```

```
      secure: false,  //false可以接触没有证书的后端服务器
      changeOrigin: true  //修改代理请求中的headers中的host，避免被检测
    }
  }
```

代码引入：绝对路径、相对路径、模块路径resolve

```
resolve:{帮助webpack找到模块代码，再不必写文件后缀名，传入文件夹自动找里面的mainFiles or index
    extensions:[".js", ".json", ".mjs", ".vue", ".ts", ".jsx", ".tsx"],预置文件后缀
    alias:{//路径别名，方便书写和查
      "@": path.resolve(__dirname, "./src"),
      "js": path.resolve(__dirname, "./src/js")
    }
  },
```

HMR原理：



8.webpack的开发&打包分离

分成三个文件，一个存都需要的配置，dev、prod配置分开，详情看文档

webpack.comm.config

webpack.dev.config

webpack.prod.config，通过merge()将common导入dev prod Config进行合并。注意文件路径../

修改package.json

```
"scripts": {
    "build": "webpack --config ./config/webpack.prod.config.js",
    "serve": "webpack serve --config ./config/webpack.dev.config.js"
  },
```

npm run build　　npm run serve可以分别有config

Vue-CLI脚手架

2021年11月2日 星期二 下午 9:32

1.vue-cli脚手架简介

CLI：command-Line Interface 命令行界面

内置webpack相关的配置，不需要从零开始。

2.安装 和 使用

npm install @vue/cli -g　　npm update @vue/cli -g

vue create 项目名称

Manually select features 手动选择

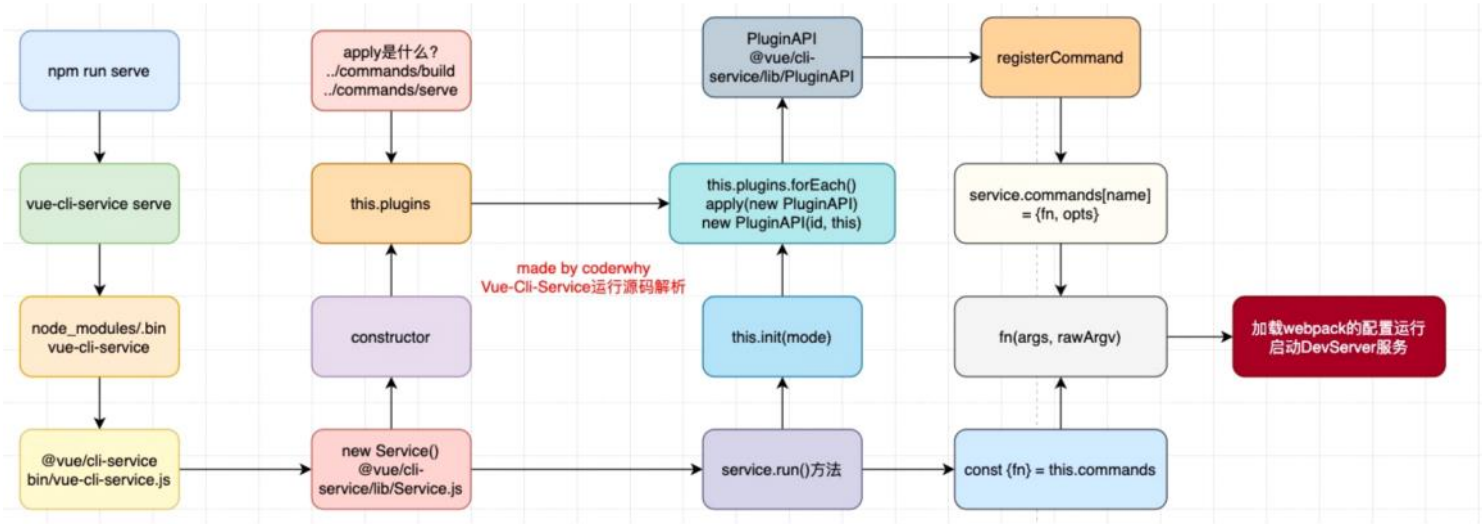
生成后的项目目录结构：

GitHub 搜 browserlist 看更多浏览器配置

.gitignore 不想上传git的文件

babel预设

3.Vue-CLI原理，有点复杂，后续再说



Vite简单看看

2021年11月4日 星期四　　上午 3:59

1.认识vite

其他工具，rollup parcel gulp vite

自称下一代开发构建工具，插件不完善

构成：基于esmodule的开发服务器，HMR极速+ 预配置rollup构建

和vue作者一样，看后续脚手架会不会迁移吧。

2.Vite的思想

先将代码转化为esmodule，让浏览器能认识，简化开发，不需要一直构建

等到需要上线，再打包

3. 安装和使用

npm install vit -g/-D

npx vite来启动项目，可以直接生成一个本地服务

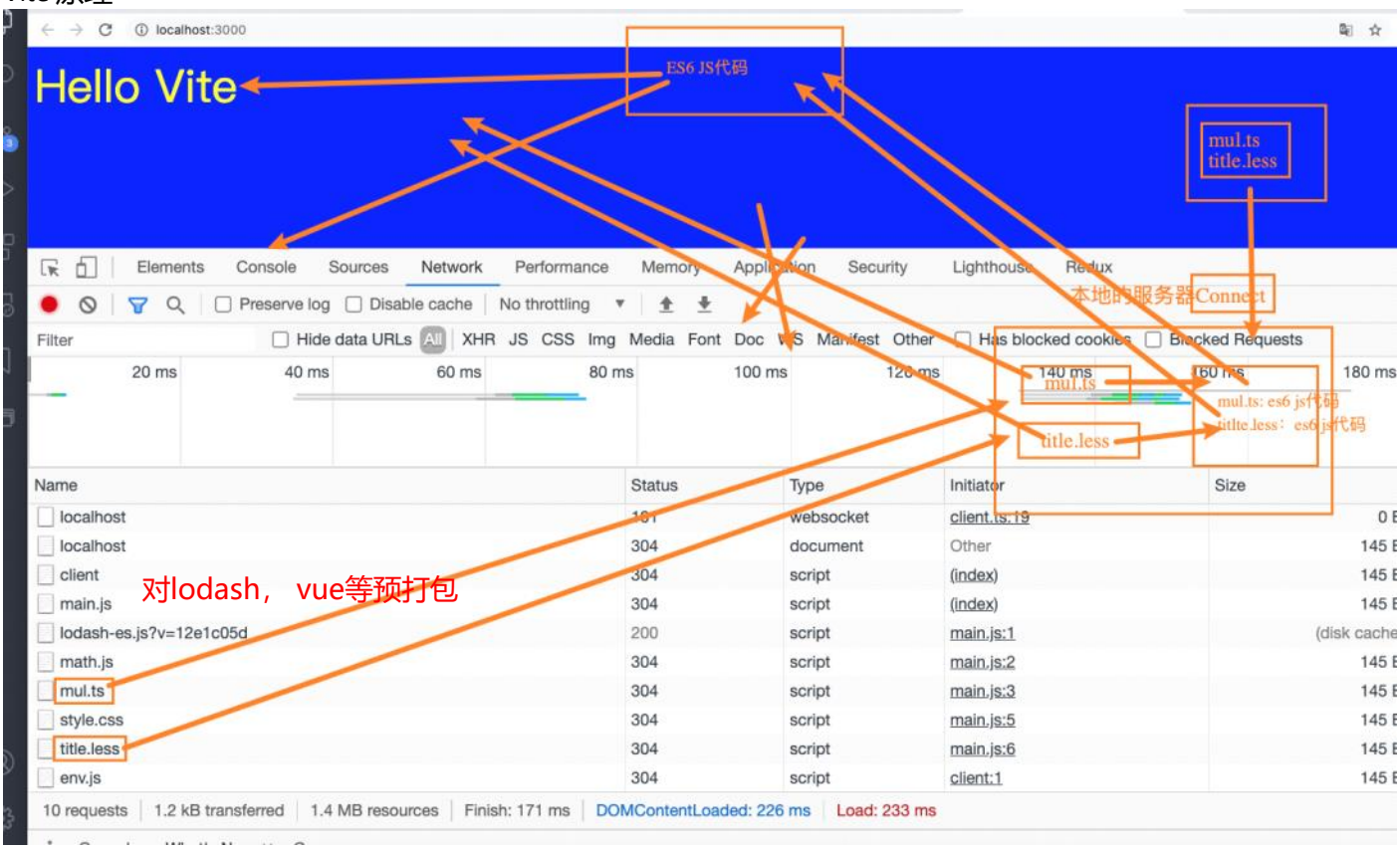
css、typescript默认支持，但是less postcss需要提前安装

npm install less/postcss

Npm install postcss-preset-env -D + 对postcss.config.js配置

```
module.exports = {
  plugins: [
    require('postcss-preset-env')
  ]
}
```

Vite 原理



用Connect做请求，内部转发，请求.ts .css .less 实际是转成es6的ts css，

对Vue的支持

- vite对vue提供第一优先级支持：
  - Vue 3 单文件组件支持：@vitejs/plugin-vue
  - Vue 3 JSX 支持：@vitejs/plugin-vue-jsx
  - Vue 2 支持：underfin/vite-plugin-vue2
- 安装支持vue的插件：

```
npm install @vitejs/plugin-vue -D
```

- 在vite.config.js中配置插件：

```
import vue from '@vitejs/plugin-vue';

module.exports = {
  plugins: [
    vue()
  ]
}
```

4.Vite 打包项目

npx vite build 打包
测试打包 npx vite preview预览打包后的效果

可以package配置一下，更快的命令 比如npm run build

5. ESBuild的速度原因
Js -> AST -> 字节码 ->机器码
esbuild是Go写的， go -> AST ->机器码
esbuild充分利用CPU多内核，饱和运行
无任何依赖，自己从零运行

6. vite脚手架

- 在开发中，我们不可能所有的项目都使用vite从零去搭建，比如一个react项目、Vue项目；
  - 这个时候vite还给我们提供了对应的脚手架工具；
- 所以Vite实际上是有两个工具的：
  - vite：相当于是一个构件工具，类似于webpack、rollup；
  - @vitejs/create-app：类似vue-cli、create-react-app；
- 如果使用脚手架工具呢？

```
npm init @vitejs/app
```

去package.json看命令

- 上面的做法相当于省略了安装脚手架的过程：

```
npm install @vitejs/create-app -g

create-app
```