开发环境调优

Webpack 开发效率插件

webpack-dashboard

每次构建结束在控制台输出一些打包相关信息,但是不直观 webpack-dashboard更好的展示这些信息 使用:

- 1. 作为插件添加到webpack配置中
- 2. 修改webpack启动方式 webpack-dashboard 替换 webpack/webpack-dev-server,并将原来的启动命令作为参数传给它

```
// package.json
{
    ...
    "scripts": {
        "dev": "webpack-dev-server"
    }
}
```

加上 webpack-dashboard 后则变为:

```
// package.json
{
    ...
    "scripts": {
        "dev": "webpack-dashboard -- webpack-dev-server"
    }
}
```

webpack-merge

配置多种打包环境,多个配置的时候,可以把公共部分提取到webpack.common.js可以使用webpack-merge

speed-measure-webpack-plugin

SMP 可以分析打包过程各个loader和plugin上消耗的时间 找出构建过程的性能瓶颈

size-plugin

模块热替换

Hot Module Replacement, HMR

开启HMR

需要手动开启,且项目必须基于webpack-dev-server/webpack-dev-middle 开发

```
const webpack = require('webpack');
module.exports = {
   // ...
   plugins: [
     new webpack.HotModuleReplacementPlugin()
   ],
   devServer: {
     hot: true,
   },
};
```

为每个模块绑定一个module.hot对象 对象包含HMR的API,可以实现特定模块 开启/关闭 HMR 可以添加逻辑,确保热更后代码正常工作

调用HMR API 可以手动,也可以借助工具 react-hot-loader vue-loader

HMR API代码应该放在入口。这样HMR对整个index依赖都会生效,发现有模块变动时,HMR会在当前浏览器环境重新执行一遍index,页面本身不会刷新

建议使用第三方HMR方案

HMR原理

HMR资源体积会大很多,因为注入了很多为了实现HMR的代码

• 本地开发

webpack-dev-server 相当于是服务端 HMR核心就是客户端从服务端拉取更新后的资源 准确的说,拉去的是chunk diff, chunk更新的部分

WDS与浏览器维护了一个websocket,本地资源发生变化,WDS向浏览器推送更新时间,附上hash,让客户端与上一次资源比较

为什么要带hash

因为文件的更改,不代表构建结果的更改。(压缩后结果可能是一样的)



客户端发现构建不同后,则会向WDS发起请求,获取更新文件的列表

```
▼ General

Request URL: http://localhost:3000/dist/2410644c20694722f9ca.hot-update.json

Request Method: GET

Status Code: ■ 200 0K

Remote Address: 127.0.0.1:3000

Referrer Policy: no-referrer-when-downgrade
```

图 9-5 请求 chunk 地址

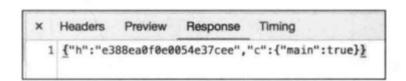


图 9-6 WDS 向浏览器的返回值

然后客户端根据返回信息,进行更新

总结

Webpack周边插件 HMR