

vite 与 webpack 操作使用

vite 与 webpack 区别:

1. vite服务器启动速度比webpack快, 由于vite启动的时候不需要打包, 也就无需分析模块依赖、编译, 所以启动速度非常快。当浏览器请求需要的模块时, 再对模块进行编译, 这种按需动态编译的模式, 极大缩短了编译时间, 当项目越大, 文件越多时, vite的开发时优势越明显。vite热更新比webpack快, vite在HRM方面, 当某个模块内容改变时, 让浏览器去重新请求该模块即可, 而不是像webpack重新将该模块的所有依赖重新编译。

Vite的使用简单, 只需执行初始化命令, 就可以得到一个预设好的开发环境, 开箱即获得一堆功能, 包括: CSS预处理、html预处理、异步加载、分包、压缩、HMR等。使用复杂度介于Parcel和Webpack的中间, 只是暴露了极少数的配置项和plugin接口, 既不会像Parcel一样配置不灵活, 又不会像Webpack一样需要了解庞大的loader、plugin生态, 灵活适中、复杂度适中。

2. webpack: 分析依赖=> 编译打包=> 交给本地服务器进行渲染。首先分析各个模块之间的依赖, 然后进行打包, 在启动webpack-dev-server, 请求服务器时, 直接显示打包结果。webpack打包之后存在的问题: 随着模块的增多, 会造成打出的 bundle 体积过大, 进而会造成热更新速度明显拖慢。

vite: 启动服务器=> 请求模块时按需动态编译显示。是先启动开发服务器, 请求某个模块时再对该模块进行实时编译, 因为现代浏览器本身支持ES-Module, 所以会自动向依赖的Module发出请求。所以vite就将开发环境下的模块文件作为浏览器的执行文件, 而不是像webpack进行打包后交给本地服务器。

分析了webpack和vite的打包方式后, 也就明白了为什么vite比webpack打包快, 因为它在启动的时候不需要打包, 所以不用分析模块与模块之间的依赖关系, 不用进行编译。这种方式就类似于我们在使用某个UI框架的时候, 可以对其进行按需加载。同样的, vite也是这种机制, 当浏览器请求某个模块时, 再根据需要对模块内容进行编译。按需动态编译可以缩减编译时间, 当项目越复杂, 模块越多的情况下, vite明显优于webpack。

热更新方面, 效率更高。当改动了某个模块的时候, 也只用让浏览器重新请求该模块, 不需要像webpack那样将模块以及模块依赖的模块全部编译一次。

优势:

- vite开发阶段, 打包快。
- vite相关生态没有webpack完善, vite可以作为开发的辅助。

vite 使用

```
1. 使用 NPM
npm init vite@latest
2. 使用 Yarn
$ yarn create vite
3. 使用 PNPM
$ pnpm create vite
```

还可以通过附加的命令行选项直接指定项目名称和你想要使用的模板。例如, 要构建一个Vite + Vue项目, 运行: