

Webpack与Vite

1.什么是webpack

什么是loader——扩展能力边界

什么是plugin——基于已有能力做一些事情

2.Vite

2.1什么是vite

2.2 vite的基石——浏览器支持

3.比较

4.为什么不舍弃webpack使用vite

5.如何理解es6的import语法之前，所有按需加载模块化方案都是假按需加载

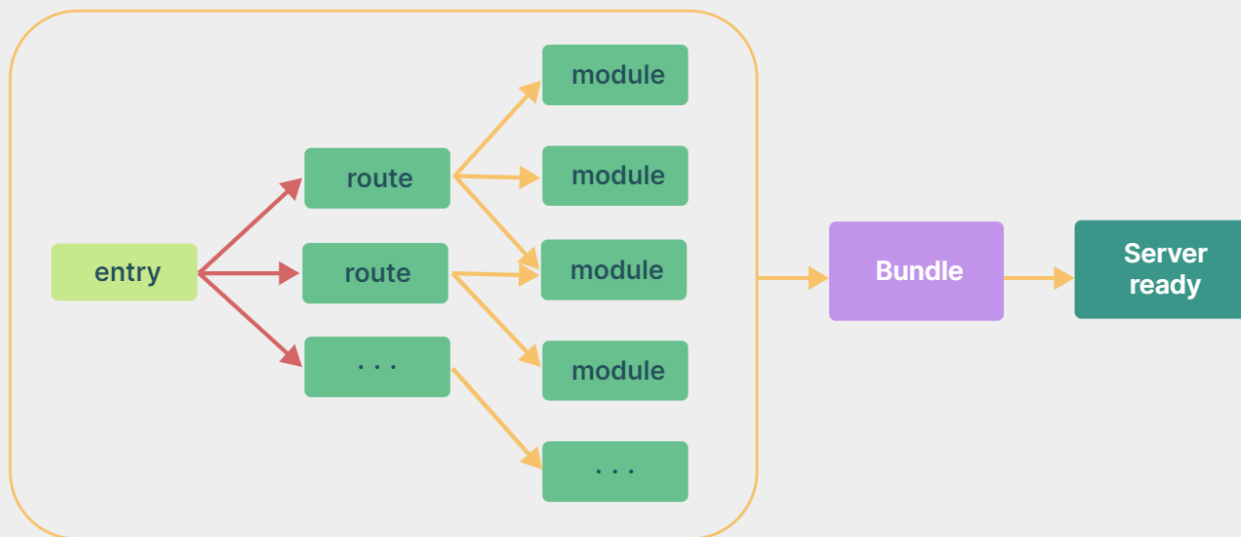
1.什么是webpack



本质上，**webpack** 是一个用于现代 JavaScript 应用程序的 *静态模块打包工具*。当 webpack 处理应用程序时，它会在内部从一个或多个入口点构建一个 [依赖图\(dependency graph\)](#)，然后将你项目中所需的每一个模块组合成一个或多个 *bundles*，它们均为静态资源，用于展示你的内容。

缺点：当我们开始构建越来越大型的应用时，需要处理的 JavaScript 代码量也呈指数级增长。包含数千个模块的大型项目相当普遍。基于 JavaScript 开发的工具就会开始遇到性能瓶颈：通常需要很长时间（甚至是几分钟！）才能启动开发服务器，即使使用模块热替换（HMR），文件修改后的效果也需要几秒钟才能在浏览器中反映出来。如此循环往复，迟钝的反馈会极大地影响开发者的开发效率和幸福感。

Bundle based dev server



什么是loader——扩展能力边界


loader 用于对模块的源代码进行转换。loader 可以使你在 import 或 "load(加载)" 模块时预处理文件。因此，loader 类似于其他构建工具中“任务(task)”，并提供了处理前端构建步骤的得力方式。loader 可以将文件从不同的语言（如 TypeScript）转换为 JavaScript 或将内联图像转换为 data URL。loader 甚至允许你直接在 JavaScript 模块中 import CSS 文件！

我们常见的loader：sass-loader, less-loader,vue-loader, ts-loader, babel-loader, html-loader

 [Loaders | webpack](#)

▼ webpack中的loader配置

JavaScript

 复制代码

```
1 module.exports = {
2   module: {
3     rules: [
4       { test: /\.css$/, use: 'css-loader' },
5       { test: /\.ts$/, use: 'ts-loader' },
6     ],
7   },
8 };
```

什么是plugin——基于已有能力做一些事情

插件 是 webpack 的 [支柱](#) 功能。Webpack 自身也是构建于你在 webpack 配置中用到的 [相同](#) 的插件系统 之上！

插件目的在于解决 [loader](#) 无法实现的[其他事](#)。Webpack 提供很多开箱即用的 [插件](#)。

[Webpack Bundle Analyzer](#)可以可视化webpack输出文件的大小，使用交互式、可缩放的树形图1；[offline-plugin](#)旨在为webpack项目提供离线体验1；[webpack-pwa-manifest](#)可以帮助你创建PWA应用程序的manifest1；[imagemin-webpack-plugin](#)可以帮助你压缩图片1。

vue中的plugin配置

JavaScript

复制代码

```
1 chainWebpack: config => {
2     config.plugin("smp").use(new SpeedMeasurePlugin());
3     config.module
4         .rule('images')
5         .test(/\.(jpe?g|png|gif|svg)$/i)
6         .use('url-loader')
7         .loader('url-loader')
8         .options({
9             limit: 20 * 1024,
10            fallback: {
11                loader: 'file-loader', options: {
12                    name: 'img/[name].[hash:8].[ext]'
13                }
14            }
15        })
16        .end()
17    },
```

2.Vite

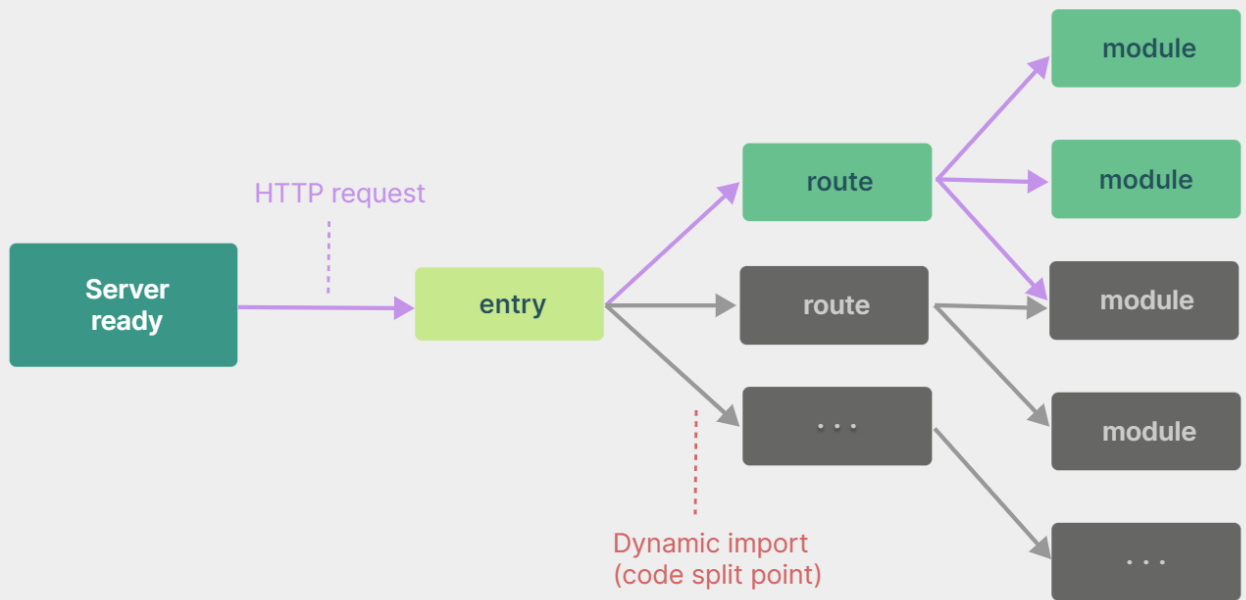


2.1什么是vite

Vite（法语意为“快速的”，发音 /vit/，发音同“veet”）是一种新型前端构建工具，能够显著提升前端开发体验。它主要由两部分组成：

- 一个开发服务器，它基于 [原生 ES 模块](#) 提供了 [丰富的内建功能](#)，如速度快到惊人的 [模块热更新 \(HMR\)](#)。
- 一套构建指令，它使用 [Rollup](#) 打包你的代码，并且它是预配置的，可输出用于生产环境的高度优化过的静态资源。

Native ESM based dev server



2.2 vite的基石——浏览器支持

默认的构建目标是能支持 [原生 ESM 语法的 script 标签](#)、[原生 ESM 动态导入](#) 和 [import.meta](#) 的浏览器。传统浏览器可以通过官方插件 [@vitejs/plugin-legacy](#) 支持 —— 查看 [构建生产版本](#) 章节获取更多细节。

3.比较

- 1.为什么我们可以在vue中使用require引入资源项目，但是vite中就得import不行
- 2.为什么vite的构建速度比webpack快许多？（按需加载，使用go编写的esbuild
- 3.为什么vite的快速热更新更快？（webpack需要对修改的模块和受影响的模块进行重新编译和打包，vite只需要将改动的模块重新提供给浏览器，让浏览器自行解析和执行。vite会将一个vue文件切割为 template, style, script模块。

1. 在 Vite 中，HMR 是在原生 ESM 上执行的。当编辑一个文件时，Vite 只需要精确地使已编辑的模块与其最近的 HMR 边界之间的链失活[\[1\]](#)（大多数时候只是模块本身），使得无论应用大小如何，HMR 始终能保持快速更新。
2. Vite 同时利用 HTTP 头来加速整个页面的重新加载（再次让浏览器为我们做更多事情）：源码模块的请求会根据 304 Not Modified 进行协商缓存，而依赖模块请求则会通过 Cache-Control: max-age=31536000,immutable 进行强缓存，因此一旦被缓存它们将不需要再次请求。

4.为什么不舍弃webpack使用vite

尽管 Vite 在开发过程中比 Webpack 快，但它们两者都有各自的优缺点。Webpack 是一个成熟且功能强大的构建工具，它拥有大量的插件和加载器，可以实现各种复杂的构建场景，例如代码分割、按需加载、CSS 预处理器等。此外，Webpack 也有着广泛的社区支持和丰富的文档。

而 Vite 是一个相对较新的构建工具，它的插件和加载器相对较少，但是提供了一些基本的插件，例如 Vue.js、React 等框架的支持。**Vite 更加适合轻量级的项目和快速迭代的开发场景。**

一个成熟且功能强大的构建工具，那么 Webpack 可能是一个不错的选择

如果您需要一个快速且现代的开发体验，那么 Vite 可能更适合您。

5.如何理解es6的import语法之前，所有按需加载模块化方案都是假按需加载

在ES6之前，JavaScript并没有原生的模块系统，因此开发人员使用CommonJS和AMD等非标准化的解决方案来实现模块化。这些方案并不是真正的按需加载，因为它们需要在运行时解析依赖关系，而不是在编译时。

ES6的import语法提供了一种静态的、声明式的方式来定义模块之间的依赖关系。这使得模块可以在运行之前被静态分析和链接，从而实现真正的按需加载。