

# 下一代构建工具 Vite 实践与探索

张毅



## 企业级一站式数字技术学习平台



原创精品  
课程



知识技能  
图谱



岗位能力  
模型



测学考评  
体系



分层分级  
培训



数字管理  
系统

数字化专业人才培养方案定制



13167596032

<https://b.geekbang.org/>



扫码免费咨询

# 关于我

- 张毅，华为技术有限公司 / 开源能力中心
- Vue 社区 Member
- Vite 社区 Collaborator
- Vue Shenzhen Meetup 发起人

# 目录

- Vite 原理与关键技术
- Vite 项目实战
- Vite 插件生态
- Vite Module Federation 探索

# Vite 是什么？



## 下一代前端开发与构建工具

Vite (法语意为 "快速的", 发音 /vit/)

- 开发环境：
  - No-Bundle 开发服务器
- 生产构建：
  - 预配置的 Rollup 打包命令，针对生产环境高度优化
- 以及
  - HMR 热更新
  - 依赖预构建
  - 内置 SSR 支持

```
$ npm create vite
```

```
$ vite
```

```
$ vite build
```



# 与 Snowpack、Webpack 对比

- It is NOT Vite' s goal to completely replace webpack !
- 目前 Snowpack 作者已将其交给社区维护，新项目 Astro 也迁移至 Vite

	Vite	Snowpack	Webpack
开发环境	No-Bundle (CJS→ESM)	No-Bundle (CJS→ESM)	Bundle (CJS/UMD/ESM)
生产构建	Rollup	可选 Webpack/Rollup/ESBuild	Webpack
插件生态	★★★★☆	★★★★☆	★★★★★
易用性	★★★★★	★★★★☆	★★★★☆
dev 启动速度	快	快	慢（项目规模越大越慢）
npm 月下载量	1085.63K +19.05%	36.59K -19.08%	71.94M +1.66%
Git Star	32.61k	19.41k	59.57k

# Vite 生态进展



⚡ 14. Vite Satisfaction score 9.6 ❤️  
From @jamstackconf Community Survey 2020—2021  
[jamstack.org/survey/2021/#c...](https://jamstack.org/survey/2021/#c...)

翻译推文



Evan You  
@youyuxi

Vite really is turning out to be more than I expected: it's now being used with not just Vue, but React, Svelte, Solid, Marko, Astro, Shopify Hydrogen, plus integrations with Storybook, Laravel, Rails etc (some combine it with InertiaJS)...

Vite 确实超出了我的预期：它现在不仅用于 Vue，还用于 React、Svelte、Solid、Marko、Astro、Shopify Hydrogen，以及与 Storybook、Laravel、Rails 等的集成（有些将它与 InertiaJS 结合使用）...



initial commit

2020.4

V1.0

2020.11

V2.0



2021.2

V2.6.x

Now



# Vite 到底有多快? - 业界案例

 Vite ⚡ 转推了  
 Bjorn Lu @bluwyoo · 5月14日  
Just migrated a production @sveltejs app from @RollupJS to @vite\_js. The speed is amazing!

Comparison 📊

Cold start: 2m15s -> 1722ms  
Warm start: 2m15s -> 283ms  
Reload: 23s -> Instant!  
Build: 3m -> 2m22s

Best productivity upgrade for a long time ⚡

- 冷启动时间 2m15s -> 1.7s
- 热启动时间 < 300ms

- 开发速度快!

 Vite ⚡  
@vite\_js

From [chat.vitejs.dev](https://chat.vitejs.dev)  
> vite updates so fast, I'm not sure if it updated or not, so I end up hitting CTRL+R just in case. To prevent that, I created a plugin that provides visual feedback when HMR happens. Now I don't need to force a reload anymore  
[github.com/joshnuss/vite-...](https://github.com/joshnuss/vite-...)

从[chat.vitejs.dev](https://chat.vitejs.dev)  
> vite 更新如此之快, 我不确定它是否更新, 所以我最终按 CTRL+R 以防万一。为了防止这种情况, 我创建了一个插件, 在 HMR 发生时提供视觉反馈。

- 热更新时间 <100ms
- 甚至需要专门的插件

- HMR 热更新快!

From Hacker News:  
Create React App (webpack) vs Vite (esbuild) on Replit containers:

- 1 second start up time on Vite vs 15 seconds for CRA
- React.js hello world project is 234mb on CRA and only 34mb on Vite
- 1GB RAM for Vite dev server vs 3GB+ for CRA

- 内存占用从 3GB 减少到 1GB
- 磁盘占用从 234M 减少到 34M

- 资源占用少!



# Vite 到底有多快？ - 实测数据

我们迁移了 20+ 项目，一些测试数据：

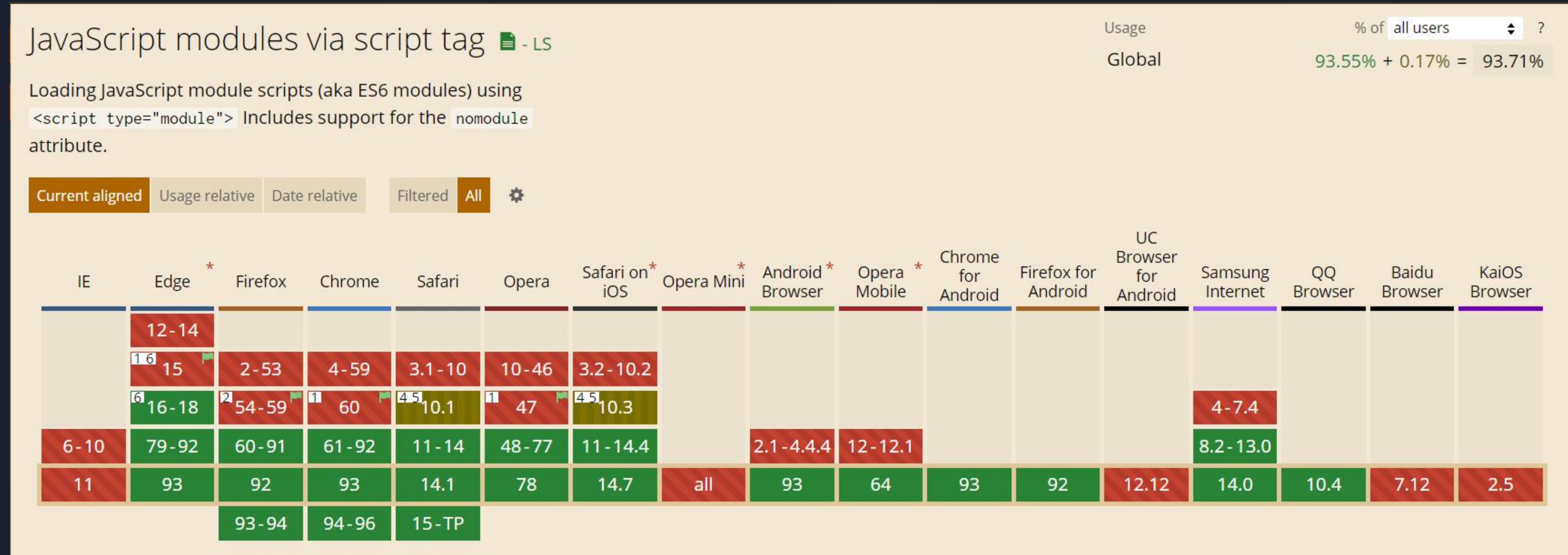
项目	代码规模	构建工具	Dev启动	页面加载时间	HMR
V***	10k	Vue Cli	10.5s	0.9s	13s
		Vite	0.6s	0.8s	0.02s
SE***	80k	Vue Cli	26.8s	0.5s	1s
		Vite	1.1s	1.2s	0.017s
Ki***	100k	Vue Cli	278s	0.9s	3.5s
		Vite	3.4s	1.6s	0.03s
We***	148k	Webpack	142s	5.8s	14s
		Vite	6.1s	8.6s	0.42s

迁移至 Vite 可获得 10~100 倍的启动性能提升，即时的模块热更新响应。

# Vite 为什么快？



# Vite 设计基于 2 个趋势：1) ES Modules 广泛支持



- 原生 ES modules 已有 93.71% 的浏览器支持率
- Internet Explorer 11 将于 2022 年 6 月 15 日停用并停止支持

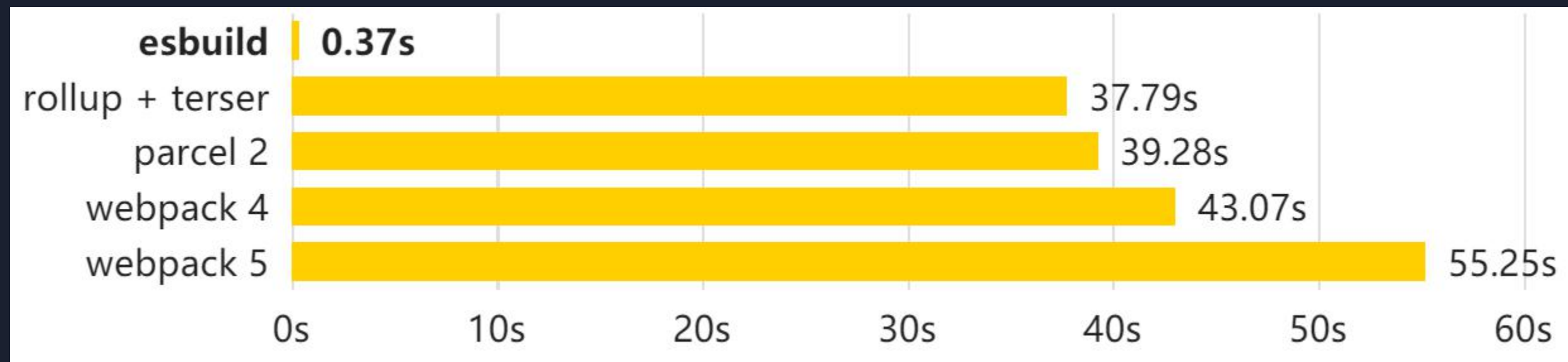


# Vite 设计基于2个趋势： 2) 高性能语言重写 Compiler

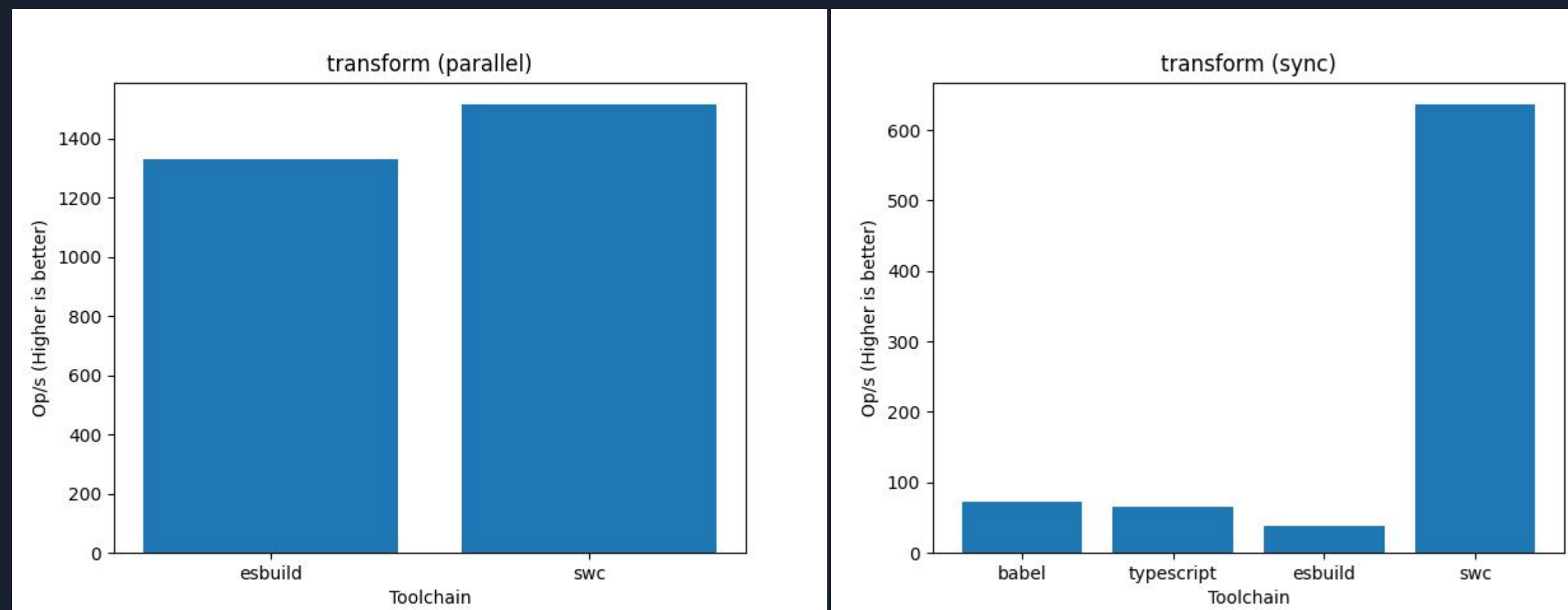


## esbuild

An extremely fast JavaScript bundler



A super-fast compiler written in rust



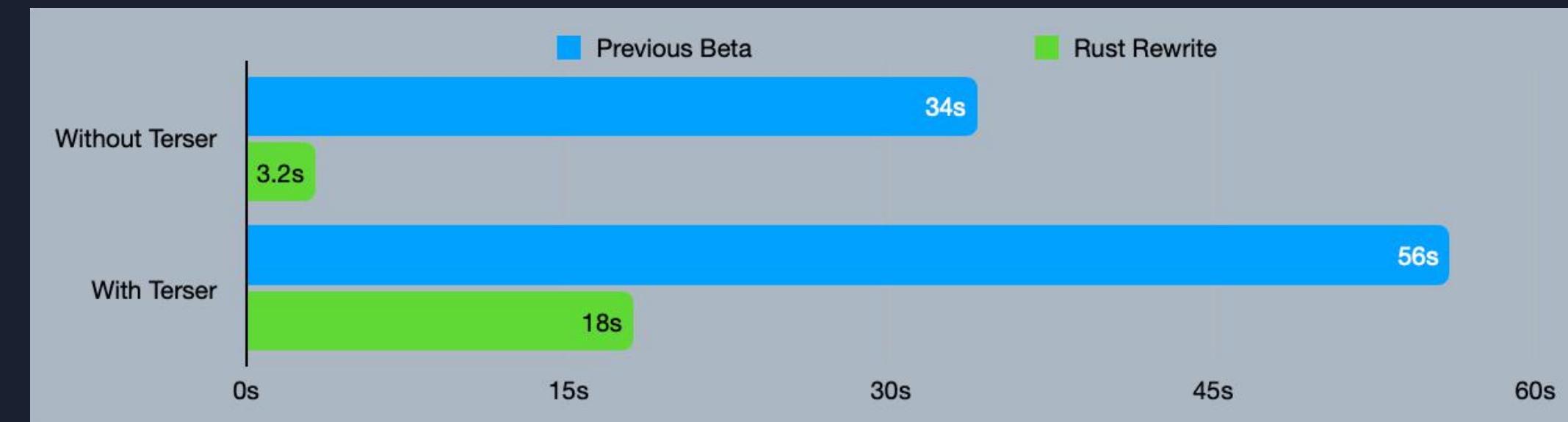
For most users, most important test result is parallel benchmark.



## PARCEL

Blazing fast, zero configuration web application bundler

A ground up rewrite of our JavaScript compiler in Rust, which improves overall build performance by up to 10x.



<https://v2.parceljs.org/blog/beta3/>



designed to replace Babel, ESLint, webpack, Prettier, Jest.

## Rome will be written in Rust



Jamie Kyle

September 21, 2021

<https://rome.tools/blog/2021/09/21/rome-will-be-rewritten-in-rust>



# Vite 核心原理 - No-Bundle

入口 index.html

```
<html lang="en">
  <body>
    <div id="app"></div>
    <script type="module" src="/src/main.ts"></script>
  </body>
</html>
```

Request URL:

<http://localhost:3000/src/main.ts>

main.ts 源文件

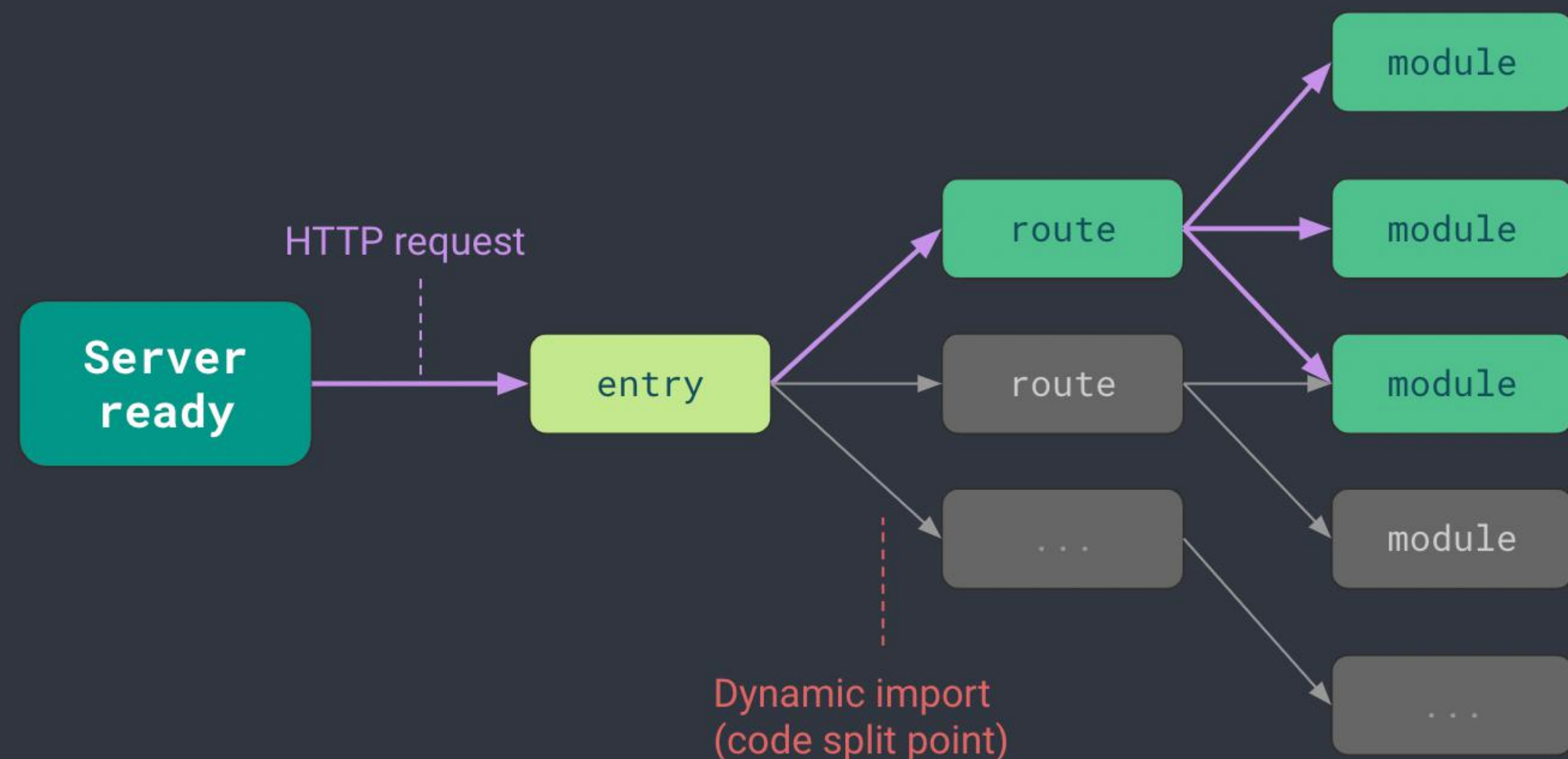
```
import { createApp } from 'vue'
import App from './App.vue'

createApp(App).mount('#app')
```

Vite 转换后

```
import { createApp } from
"/node_modules/.vite/vue.js?v=b2be7c94";
import App from "/src/App.vue";
createApp(App).mount("#app");
```

## Native ESM based dev server



# No-Bundle 带来的优缺点

- 优势：

- 快！

- ESM 标准兼容

- 带来的问题：

- 请求数量增多

- Dev 和生产不一致

- 部分依赖不提供 ESM



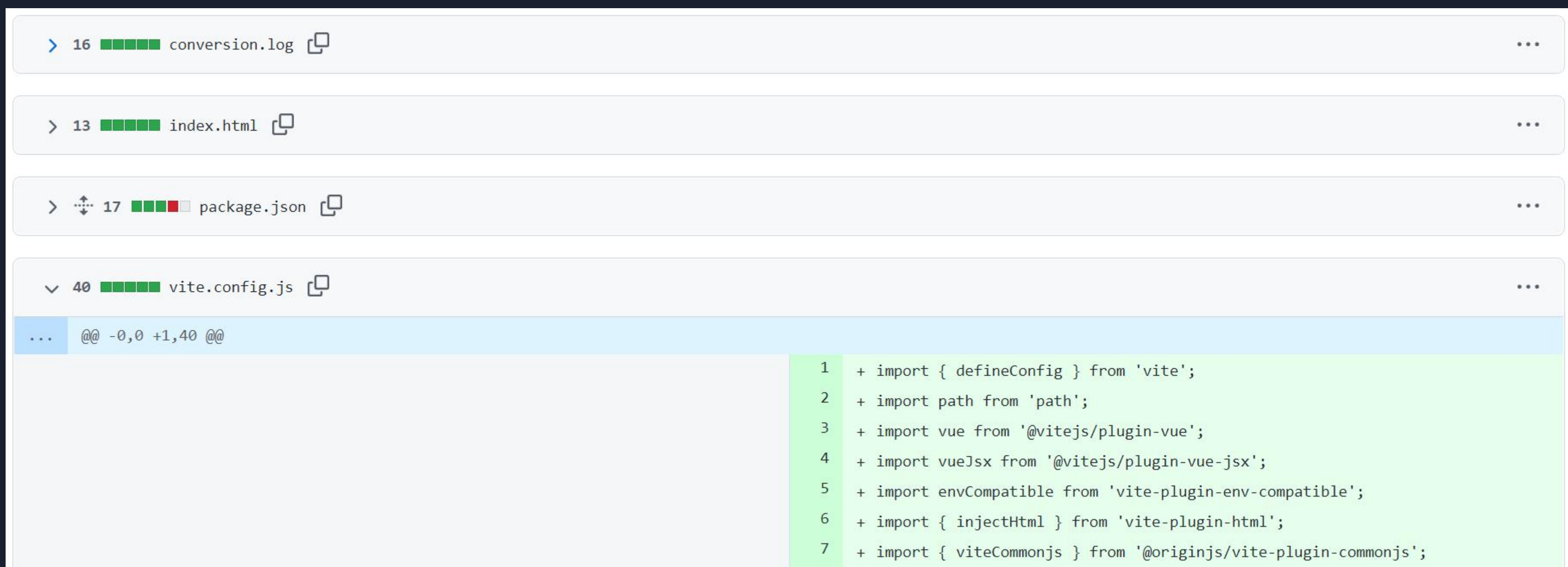
# 目录

- Vite 原理与关键技术
- Vite 项目实战
- Vite 插件生态
- Vite Module Federation 探索

# Webpack 迁移 Vite 实践

## 一键迁移工具 webpack-to-vite

- 项目地址: <https://github.com/originjs/webpack-to-vite>
- 使用方式:
  - `npx @originjs/webpack-to-vite -d <project path>`



The screenshot shows a code editor with four files: conversion.log, index.html, package.json, and vite.config.js. The vite.config.js file is selected and expanded, showing the following code:

```
... @@ -0,0 +1,40 @@  
1 + import { defineConfig } from 'vite';  
2 + import path from 'path';  
3 + import vue from '@vitejs/plugin-vue';  
4 + import vueJsx from '@vitejs/plugin-vue-jsx';  
5 + import envCompatible from 'vite-plugin-env-compatible';  
6 + import { injectHtml } from 'vite-plugin-html';  
7 + import { viteCommonjs } from '@originjs/vite-plugin-commonjs';
```



# Vite 使用常见问题 – CommonJS 的挑战

- 社区 225+ issue (占比8.6%)
- dev 环境:
  - src 源码, 手工修改为 ESM
  - node\_modules: optimizeDeps.include
  - 使用插件 @originjs/vite-plugin-commonjs
- 生产构建:
  - Vite 默认使用 @rollup/plugin-commonjs
  - 配置 build.commonjsOptions, 例如: transformMixedEsModules

CJS

```
const obj = require('./file.js');
module.exports = {
  obj
};
```



ESM

```
import obj from './file.js'
export default {
  obj
};
```

# Vite 使用常见问题 – 浏览器兼容

- dev 开发环境依赖 ESM，但 build 打包仍可兼容 IE11
- @vitejs/plugin-legacy
- 原理：
  - ✓ 使用@babel/preset-env
  - ✓ SystemJS 格式
  - ✓ <script nomodule>

# Vite 使用常见问题 – 其他

- jsx 支持

- .vue 文件必须显示声明 `<script lang= “jsx” ></script>`
- 文件后缀必须是 .jsx

- Typescript 兼容

- esbuild 只做转译，不执行类型检查
- 每个文件独立编译 isolatedModules

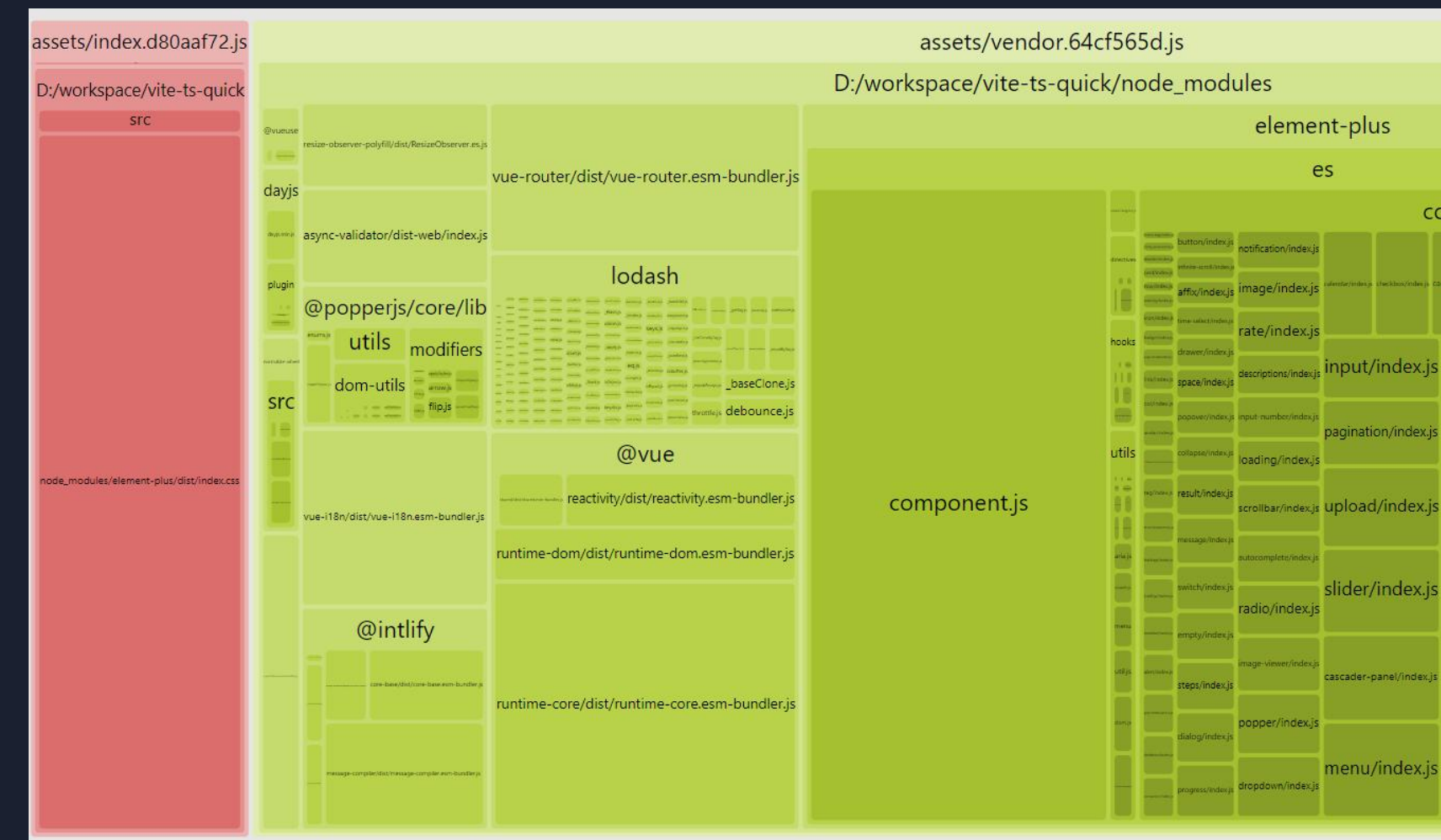
- Node built-ins

- Vite will not support (same as webpack5)



# Vite 优化 – 生产构建体积优化

- 使用插件 rollup-plugin-visualizer
- build.rollupOptions.external
- Code Splitting
  - ✓ 利用好 dynamic import
- CSS 按需加载
  - ✓ vite-plugin-style-import



```
import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'
import { visualizer } from 'rollup-plugin-visualizer'
export default defineConfig({
  plugins: [
    vue()
  ],
  build: {
    rollupOptions: {
      plugins: [ visualizer( ) ],
      external:['vue']
    }
  }
})
```

# Vite 优化 – 生产构建速度优化

- 主要是针对 rollup 的优化
  - build.rollupOptions.external ↓ 16.5s
  - build.commonjsOptions.sourceMap -> false ↓ 0.5s
  - build.brotliSize -> false ↓ 0.3s
  - build.minify -> `esbuild` 2.6.0默认 ↓ 6.2s (20~40x fast)

整体构建时间可从 29.8s 优化至 6.3, 降低 79%

是时候迁移到 Vite 了吗？



# 目录

- Vite 原理与关键技术
- Vite 项目实战
- Vite 插件生态
- Vite Module Federation 探索

# Vite 插件生态



awesome

<https://github.com/vitejs/awesome-vite>

 **Vite Rollup Plugins**

<https://vite-rollup-plugins.patak.dev>

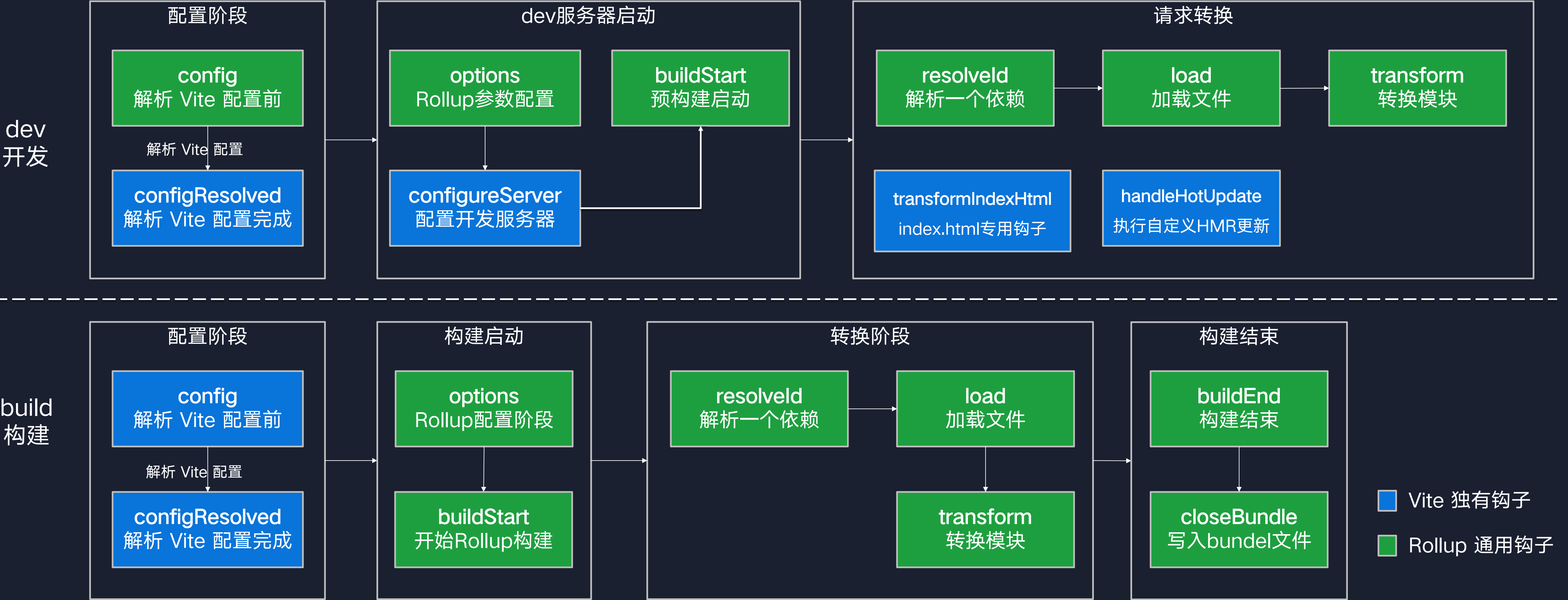
# Vite 常用插件

- @vitejs/plugin-vue / @vite-plugin-vue2
  - ✓ Vue 3 / Vue 2 支持
- @vitejs/plugin-legacy
  - ✓ 传统浏览器兼容支持
- @vitejs/plugin-vue-jsx
  - ✓ 提供 Vue 3 JSX 支持
- @vitejs/plugin-react
  - ✓ react 支持



# Vite 插件开发实践(1)

Vite 插件接口基于 Rollup 进行扩展:



# Vite 插件开发实践(2)

## • 执行顺序

➤ 通过 enforce 属性调整

➤ 插件按照以下顺序执行：

- Alias
- 带有 enforce: 'pre' 的用户插件
- Vite 核心插件
- 没有 enforce 值的用户插件
- Vite 构建用的插件
- 带有 enforce: 'post' 的用户插件
- Vite 后置构建插件（最小化，manifest，报告）

## • 情景应用

➤ 默认在 dev 和 build 模式中都会调用

➤ 使用 apply 指明仅在 'build' 或 'serve' 模式

```
export default function myPlugin() {
  return {
    name: 'my-plugin', // 插件名称
    enforce: 'pre', // 调整插件被执行顺序
    apply: "build | serve", // 指定插件应用情景
    options(options) {
    },
    buildStart(options) {
    },
    resolveId(id) {
    },
    load(id) {
    },
    transform(src, id) {
    },
    buildEnd(error) {
    },
    closeBundle() {
    },
    config(config, env) {
    },
    configResolved(config) {
    },
    configureServer(server) {
    },
    transformIndexHtml(html, ctx) {
    },
    handleHotUpdate(ctx) {
    }
  }
}
```

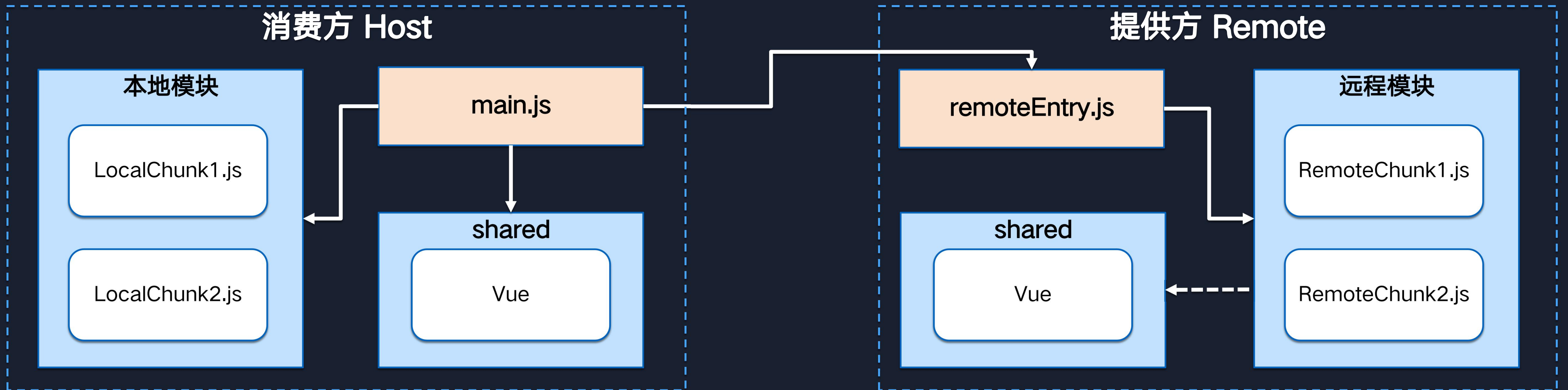
# 目录

- Vite 原理与关键技术
- Vite 项目实战
- Vite 插件生态
- Vite Module Federation 探索



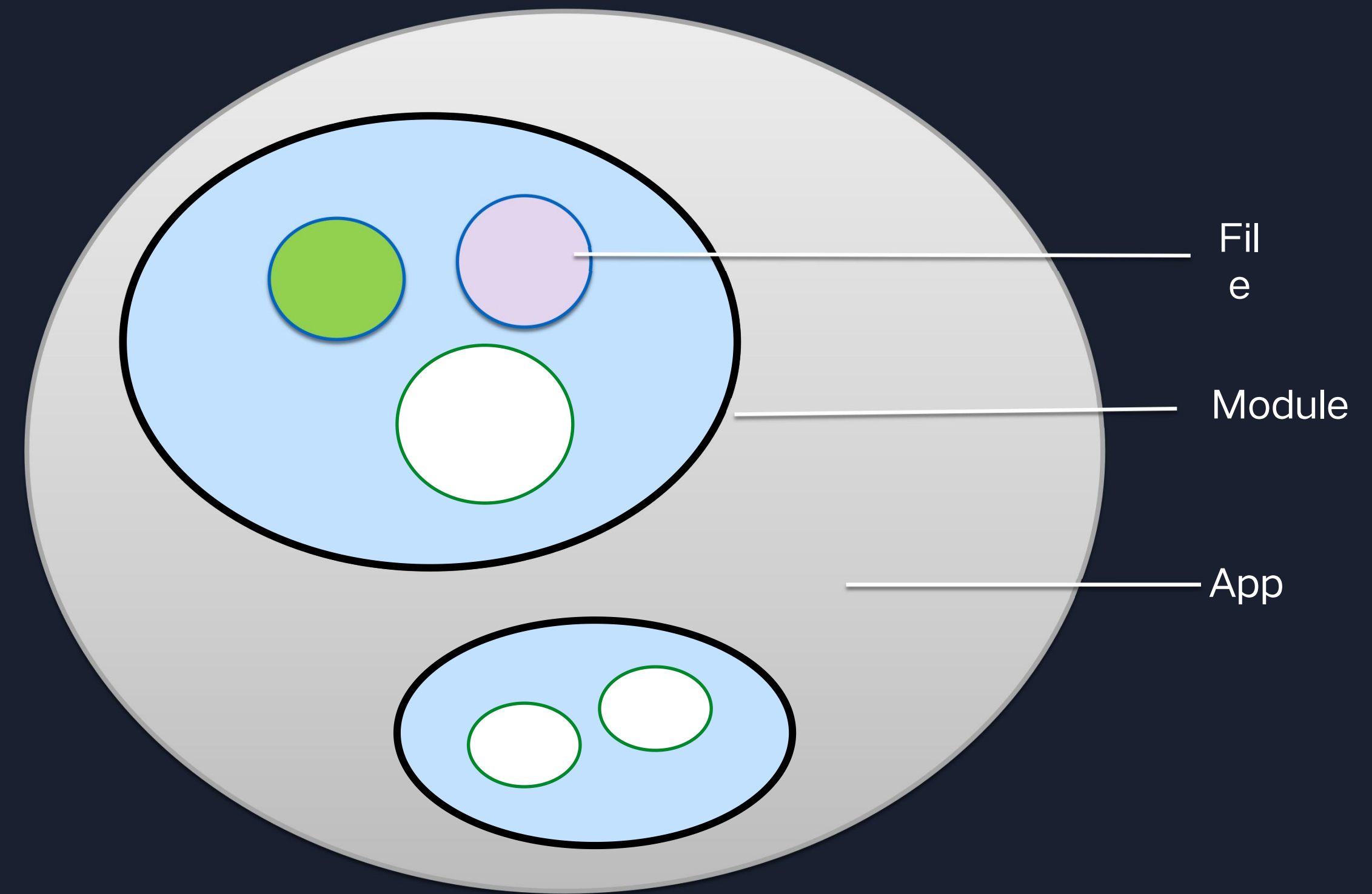
# 什么是 Module Federation

- 通过 Module Federation 可动态运行另一个远程模块的代码
- 可共享远端和本地依赖
- Bidirectional-host 去中心化



# Module Federation 与 ES Modules

- Bundle vs Unbundled
- Why Module Federation?
  - 打包粒度
  - 模块按需编译
  - 跨项目共享
  - Micro Frontend



# 基于 Vite 的 Module Federation

- 项目地址: <https://github.com/originjs/vite-plugin-federation>
- 兼容 Webpack 配置格式
- 可在 Rollup & Vite 项目使用
- 兼容 Dev 模式

```
import { defineConfig } from 'vite'
import federation from "@originjs/vite-plugin-federation";

export default defineConfig({
  plugins: [
    federation({
      name: 'module-name',
      filename: 'remoteEntry.js',
      exposes: {
        './Button': './src/Button.vue',
      },
      remotes: {
        foo: 'remote_foo'
      }
      shared: ['vue']
    })
  ],
})
```



# 总结

Vite 作为下一代前端开发与构建工具优势明显



- ✓ 极速的服务启动和热更新
- ✓ 开箱即用的丰富功能
- ✓ 优化的生产构建
- ✓ 易于开发的通用插件

还等什么？现在就用 Vite ！



# THANKS



# 为一线互联网公司核心技术 人员提供优质内容

☑ TGO专访

☑ 技术干货

☑ 每周精要

☑ 行业趋势



关注 InfoQ 公众号



# VUE SHENZHEN Meetup

- 官 网: <http://vueshenzhen.com/>
- bilibili: VueShenzhen

