

# 村长简介

**16年工作经验**

**全栈工程师、架构师**

**写过Java、C#、AS、JS等多种语言**

**精通各种前端技术**

**擅长造轮子**

**在大厂工作5年，领导过数十人开发团队**

**现在是一名职业讲师，致力于传道授业**

**Github老炮：57code**

<https://github.com/57code>

**B站新星：Young村长**

<https://space.bilibili.com/480140591>

**掘金优秀作者：杨村长**

<https://juejin.cn/user/325111174926350>

**抖音小鲜肉：前端杨村长**



WEBSITE  
Development

# VITE2项目工程化 及 原 理 剖 析

DAY1

@杨村长

# 二天学习目标

1

**DAY1 实战篇：Vite2项目工程化  
插件开发实战**

2

**DAY2 进阶篇：Vite原理剖析  
手写实现**

# 今日学习目标

## DAY1 - 实战篇：Vite2项目工程化 插件开发实战

1.快速起始

2.项目结构分析

3.静态资源加载

4.CSS使用和组织

5.TS整合

6.代码规范ESLint

7.代码测试

8.Vite2插件开发概述

9.Vite2插件机制分析

10.Vite2插件开发实战

## 2021前端会有什么新的变化?

关注问题

写回答

邀请回答

👍 好问题 109

💬 2 条评论

🔗 分享 ...

查看全部 51 个回答



尤雨溪



前端开发等 3 个话题下的优秀答主

563 人赞同了该回答

会有很多人抛弃 webpack 开始用 vite

发布于 01-07

▲ 已赞同 563



💬 129 条评论

🔗 分享

★ 收藏

♥ 喜欢



Vite是个啥

Vite是一个开发构建工具，开发过程中它利用浏览器native ES Module特性按需导入源码，预打包依赖。是为开发者量身定做的一套先进的开发工具，开发体验丝滑，默认还整合vue3，特点：

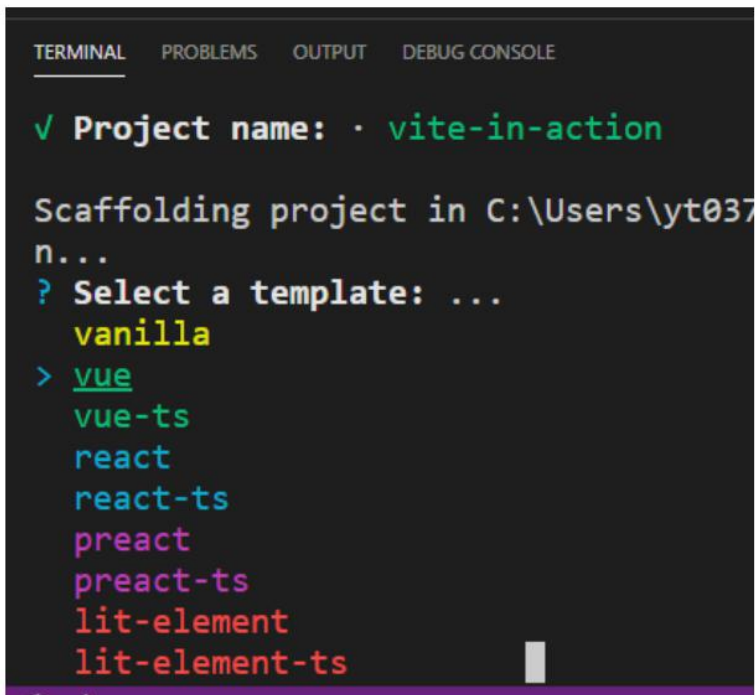
- 启动快
- 更新快



# 体验vite

创建

```
$ npm init @vite/app
```



A terminal window with tabs for TERMINAL, PROBLEMS, OUTPUT, and DEBUG CONSOLE. The terminal shows the following output:

```
✓ Project name: · vite-in-action

Scaffolding project in C:\Users\yt037
n...
? Select a template: ...
  vanilla
> vue
  vue-ts
  react
  react-ts
  preact
  preact-ts
  lit-element
  lit-element-ts
```

## 将资源引入为URL

服务时引入一个静态资源会返回解析后的公共路径:

```
import logo from './assets/logo.png' // 输出/src/assets/logo.png
```

使用这个路径

```

```



## 样式处理：

- 导入CSS
- scoped CSS
- CSS module
- CSS预处理器

# TS整合

可直接导入.ts 文件，在SFC中<script lang="ts">，ts参考配置：

```
{
  "compilerOptions": {
    "target": "esnext",
    "module": "esnext",
    "strict": true,
    "jsx": "preserve",
    "moduleResolution": "node",
    "types": ["vite/client"],
    "isolatedModules": true
  },
  "include": [
    "src/**/*.ts", "src/**/*.d.ts", "src/**/*.tsx", "src/**/*.vue", "tests/unit"
  ]
}
```

# 代理

配置服务器代理, vite.config.js

```
export default {  
  server: {  
    proxy: {  
      '/api': {  
        target: 'http://jsonplaceholder.typicode.com',  
        changeOrigin: true,  
        rewrite: path => path.replace(/^\/api/, "")  
      }  
    }  
  },  
}
```

# 数据mock

安装依赖

```
npm i mockjs -S  
npm i vite-plugin-mock -D
```

引入插件, vite.config.js

```
import {viteMockServe} from 'vite-plugin-mock'  
export default defineConfig({  
  plugins: [ viteMockServe({}) ]  
})
```

## 代码规范

我们借助 `eslint+prettier` 规范项目代码。

安装依赖，`package.json`

```
{
  "devDependencies": {
    "@typescript-eslint/eslint-plugin": "^4.15.2",
    "@typescript-eslint/parser": "^4.15.2",
    "@vue/eslint-config-prettier": "^6.0.0",
    "@vue/eslint-config-typescript": "^7.0.0",
    "@vuedx/typescript-plugin-vue": "^0.6.3",
    "eslint": "^7.20.0",
    "eslint-plugin-prettier": "^3.3.1",
    "eslint-plugin-vue": "^7.6.0",
    "prettier": "^2.2.1",
  }
}
```

```
module.exports = {
  root: true,
  env: {
    browser: true,
    es2021: true,
    node: true,
  },
  extends: [
    'plugin:vue/vue3-recommended',
    'eslint:recommended',
    '@vue/typescript/recommended',
    '@vue/prettier',
    '@vue/prettier/@typescript-eslint',
  ],
  parserOptions: {
    ecmaVersion: 2021,
  },
  plugins: [],
  rules: {
    "no-unused-vars": "off",
    "@typescript-eslint/no-unused-vars": "off"
  },
}
```

# 测试环境

利用 `jest` 和 `@vue/test-utils` 测试组件

```
"jest": "^26.6.3",  
"@types/jest": "^26.0.20",  
"vue-jest": "^5.0.0-alpha.7"  
"babel-jest": "^26.6.3",  
"@babel/preset-env": "^7.12.17",  
"@vue/test-utils": "^2.0.0-beta.9",  
"ts-jest": "^26.5.1",  
"@babel/preset-typescript": "^7.12.17",
```

配置, jest.config.js

```
module.exports = {
  transform: {
    // 用 `vue-jest` 处理 `*.vue` 文件
    '^.+\\.vue$': 'vue-jest',
    '^.+\\.jsx?$': 'babel-jest', // Adding this line solved the issue
    '^.+\\.tsx?$': 'ts-jest',
  },
  // support alias
  moduleNameMapper: {
    '^@/components(.*)$': '<rootDir>/src/components$1',
  },
  testMatch: ['**/tests/unit/**/*.[jt]s?(x)'],
}
```

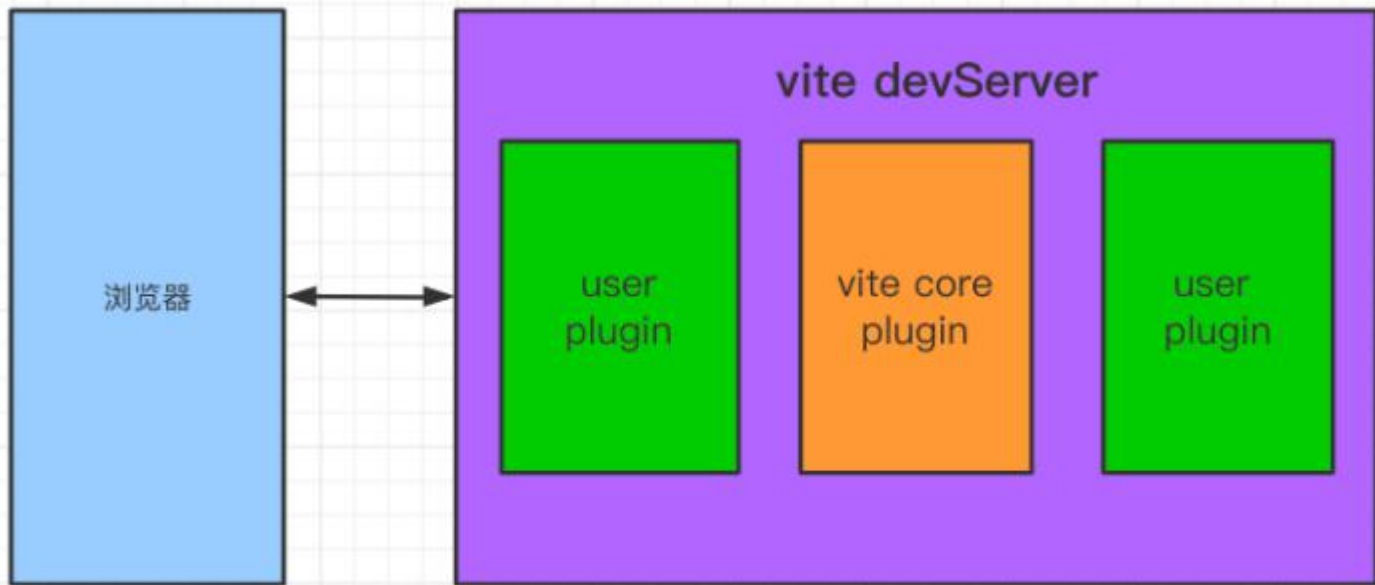


配置, babel.config.js

```
module.exports = {  
  presets: [  
    ['@babel/preset-env', { targets: { node: 'current' } }],  
    '@babel/preset-typescript',  
  ],  
}
```

# Vite插件是什么

使用Vite插件可以扩展Vite能力，比如解析用户自定义的文件输入，在打包代码前转译代码，或者查找第三方模块。



## Vite插件的形式

```
export default {  
  name: 'my-vite-plugin',  
  resolveId(id) {},  
  load(id) {},  
  transform(code) {}  
}
```

```
export default function (options) {  
  return {  
    name: "my-vite-plugin",  
    resolveId(id) {},  
    load(id) {},  
    transform(code) {},  
  };  
}
```

## 插件钩子

开发时，Vite创建一个插件容器按照顺序调用各个钩子。

- config: 修改Vite配置
- configResolved: Vite配置确认
- configureServer: 用于配置dev server
- transformIndexHtml: 用于转换宿主页
- resolveId 创建自定义确认函数，常用语定位第三方依赖
- load 创建自定义加载函数，可用于返回自定义的内容
- transform 可用于转换已加载的模块内容
- handleHotUpdate: 自定义HMR更新时调用

# 插件编写实操

## 实现一个国际化插件vite-plugin-i18n

我们希望能解析SFC中定义多语言的自定义块，App.vue：

```
<i18n>
{
  "en": {
    "language": "Language",
    "hello": "hello, world!"
  },
  "ja": {
    "language": "语言",
    "hello": "你好，世界！"
  }
}
</i18n>
```

# 进阶路线脑图

<https://frontendmasters.com/guides/learning-roadmap/>