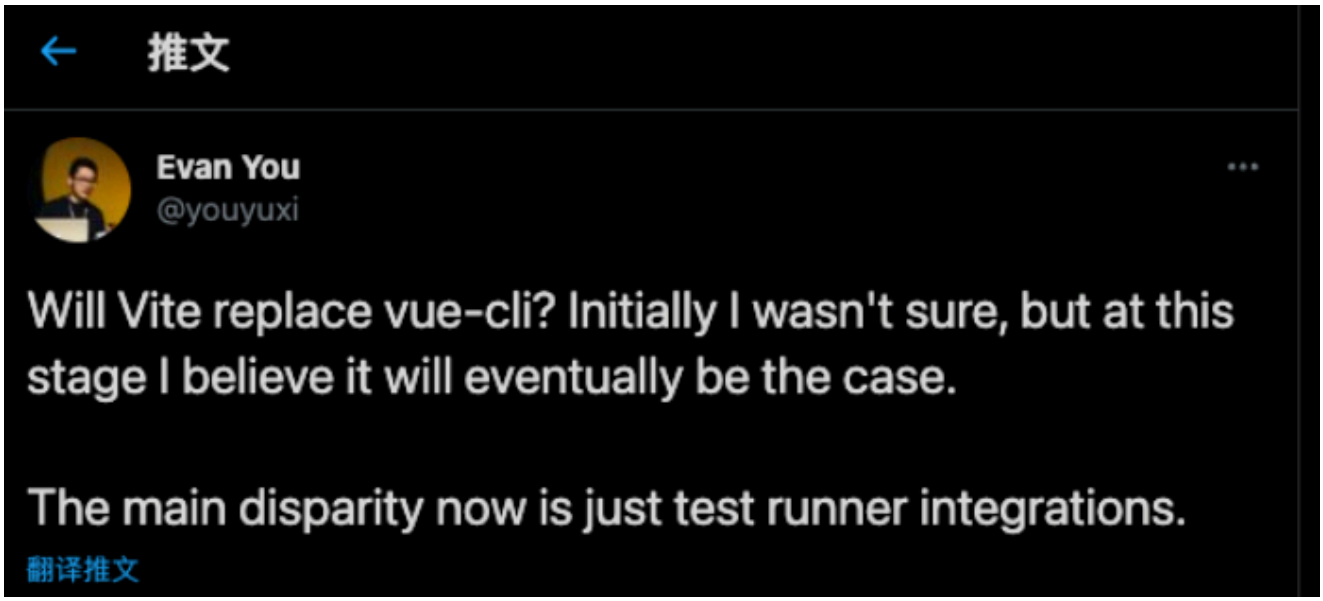
构建工具第二天 - ESModule与Vite

> 尤雨溪: Vite 会取代 vue-cli 吗?
>
> https://juejin.cn/post/6935750217924870152
>
> 
>
>
> 杨村长 Vite实战
>
> https://juejin.cn/post/6926822933721513998
>
> https://github.com/57code/vite2-in-action

- import add from './add' // 相对路径 './add'
- import vue from 'vue'   // '@module/vue' => module
- import add from './abc.vue' sfc 单文件组件 =》 js
- import style from './abc.css' css => js
- import jsx tsx
- 热更新

vite

一、Vite是什么

Vite(读音类似于[weɪt]，法语，快的意思) 是一个由原生 ES Module 驱动的 Web 开发构建工具。在开发环境下基于浏览器原生 ES imports 开发，在生产环境下基于 Rollup 打包

vite 的特点

- Lightning fast cold server start - 闪电般的冷启动速度
- Instant hot module replacement (HMR) - 即时热模块更换（热更新）

- True on-demand compilation - 真正的按需编译

要求

```
- Vite 要求项目完全由 ES Module 模块组成
- common.js 模块不能直接在 Vite 上使用
- 打包上依旧还是使用 rollup 等传统打包工具
```

安装

```
npm i vite -s
```

```
# 指定文件名和模板
npm init @vitejs/app vite-element-admin --template vue
```

Vite2主要变化

- 配置选项变化：`vue特有选项`、创建选项、css选项、jsx选项等、别名行为变化：不再要求/开头或结尾
- Vue支持：通过 @vitejs/plugin-vue插件支持
- React支持
- HMR API变化
- 清单格式变化
- 插件API重新设计

配置别名

vite.config.js

```
import path from 'path'
export default {
  resolve: {
    alias: {
      "/@": path.resolve(__dirname, "src"),
      comps: path.resolve(__dirname, "src/components"),
    },
  },
}
```

配置文件vite.config.js

```
import { defineConfig } from 'vite'
import vue from '@vitejs/plugin-vue'

// https://vitejs.dev/config/
export default defineConfig({
  plugins: [vue()]  // 插件形式支持vue
})
```

添加路由

```
npm i vue-router@next -S
```

添加视图 /views/home.vue

```
<template>
    <div>Home....</div>
</template>
```

路由配置 /router/index.js

```
import { createRouter, createWebHashHistory } from 'vue-router';

const router = createRouter({
  history: createWebHashHistory(),
  routes: [
    { path: '/', component: () => import('/@/views/home.vue') }
  ]
});

export default router
```

引入 main.js

```
import router from "/@/router";
createApp(App)
.use(router) // 添加路由插件
.mount("#app");
```

修改布局 main.js

```
<template>
  <router-view></router-view>
</template>
```

状态管理

```
npm i vuex@next -S
```

Store配置，`store/index.js`

```
import { createStore } from "vuex";

export default createStore({
  state: {
    count: 0,
  },
  mutations: {
    increment(state) {
      state.count++;
    },
  },
});
```

引入，`main.js`

```
import store from "@/store";
createApp(App).use(store).mount("#app");
```

使用状态 /views/home.vue

```
<div>{{ $store.state.count }}</div>
<button @click="$store.commit('increment')">Add</button>
```

二、Vite原理分析

1. EsModule

服务器端

```
const Koa = require('koa')
const app = new Koa()
app.use(async (ctx) => {
  const {
    request: { url, query },
  } = ctx;
  console.log("url:" + url, "query type", query.type);
  // 首页
  if (url == "/") {
    ctx.type = "text/html";
    let content = fs.readFileSync("./index.html", "utf-8");
    ctx.body = content;
  }
})
app.listen(3000, () => {
  console.log('Vite Start ....')
})
```

新建页面index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <link rel="icon" href="/favicon.ico" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Vite App</title>
</head>
<body>
  <h1>然叔 666</h1>
  <div id="app"></div>
  <script>
  </script>
  <script type="module" src="/src/main.js"></script>
</body>
```

```
</html>
```

新建/src/main.js

```
console.log('main ....')
```

添加模块解析 /index.js

/src/moduleA

```
export const str = "Hello Vite";
```

/src/main.js

```
import { str } from "./moduelA.js";
console.log(str);
```

```
else if (url.endsWith(".js")) {
    // js文件
    const p = path.resolve(__dirname, url.slice(1));
    ctx.type = "application/javascript";
    const content = fs.readFileSync(p, "utf-8");
   ctx.body = content
  }
```

添加依赖解析

> From ('./xxxx') => from ('./xxx')
>
> From ('yyyy') => from ('/@modules/yyyy')

```
function rewriteImport(content) {
  return content.replace(/ from ['|"]([^'"]+)['|"]/g, function (s0, s1) {
    console.log("s", s0, s1);
    // . ../ /开头的，都是相对路径
    if (s1[0] !== "." && s1[1] !== "/") {
      return ` from '/@modules/${s1}'`;
    } else {
      return s0;
    }
  });
}
// 添加模块改写
 ctx.body = rewriteImport(content);
```

第三方依赖支持

/src/main.js

```
import { createApp, h } from "vue";
const App = {
  render() {
    return h("div", null, [h("div", null, String("123"))]);
  },
};
createApp(App).mount("#app");
```

```
else if (url.startsWith("/@modules/")) {
    // 这是一个node_module里的东西
    const prefix = path.resolve(
      __dirname,
      "node_modules",
      url.replace("/@modules/", "")
    );
    const module = require(prefix + "/package.json").module;
    const p = path.resolve(prefix, module);
    const ret = fs.readFileSync(p, "utf-8");
    ctx.type = "application/javascript";
    ctx.body = rewriteImport(ret);
  }
```

```
if (url == "/") {
    ctx.type = "text/html";
    let content = fs.readFileSync("./index.html", "utf-8");
    // 添加
    content = content.replace(
        "<script ",
        `
        <script>
          window.process = {env:{ NODE_ENV:'dev'}}
        </script>
        <script
      `
      );
    ctx.body = content;
  }
```

SFC组件支持

App.vue

```
<template>
  <h1>大家好 然叔666</h1>
  <h2>
    <span>count is {{count}} *2={{double}}</span>
    <button @click="count++">戳我</button>
  </h2>
</template>

<script>
import {ref,computed} from 'vue'
export default {
  setup(){
    const count = ref(6)
    function add(){
      count.value++
    }
    const double = computed(()=>count.value*2)
    return {count,add,double}
  }
}
</script>
```

main.js

```
import { createApp } from 'vue' // node_module
import App from './App.vue' // 解析成额外的 ?type=template请求
import './index.css'

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

index.css

```
h1{
  color:red;
}
```

index.js

```
const compilerSfc = require("@vue/compiler-sfc"); // .vue
const compilerDom = require("@vue/compiler-dom"); // 模板

  else if (url.endsWith(".css")) {
    const p = path.resolve(__dirname, url.slice(1));
    const file = fs.readFileSync(p, "utf-8");
    const content = `
    const css = "${file.replace(/\n/g, "")}"
    let link = document.createElement('style')
    link.setAttribute('type', 'text/css')
    document.head.appendChild(link)
    link.innerHTML = css
    export default css
    `;
    ctx.type = "application/javascript";
    ctx.body = content;
  } else if (url.indexOf(".vue") > -1) {
    // vue单文件组件
    const p = path.resolve(__dirname, url.split("?")[0].slice(1));
    const { descriptor } = compilerSfc.parse(fs.readFileSync(p, "utf-8"));

    if (!query.type) {
      ctx.type = "application/javascript";
      // 借用vue自导的compile框架 解析单文件组件，其实相当于vue-loader做的事情
      ctx.body = `
  ${rewriteImport(
    descriptor.script.content.replace("export default ", "const __script = ")
  )}
  import { render as __render } from "${url}?type=template"
  __script.render = __render
  export default __script
      `;
    } else if (query.type === "template") {
      // 模板内容
      const template = descriptor.template;
      // 要在server端吧compiler做了
      const render = compilerDom.compile(template.content, { mode: "module" })
        .code;
      ctx.type = "application/javascript";

      ctx.body = rewriteImport(render);
    }
  }
```