说起尤雨溪的Vue，前端开发者应该都耳熟能详，Vue这个”后起之秀“已和React、AngularJS并列为前端三大主流开发框架之一，在国内尤其受到欢迎。最近尤雨溪又推出了Vite，这使得webpack开发者直喊大哥.....

那么Vite是什么呢？它是基于Native ES imports的开发服务器。利用浏览器去解析imports，在服务端按需编译返回，跳过了打包的概念。服务器随启随用，支持热更新。针对生产环境则可以用相同代码rollup打包。最新的Vite 2.0与框架无关，它通过@vitejs/plugin-vue和@vitejs/plugin-react-refresh来分别支持Vue和React等不同的开发框架。

建立Vite+Vue3模板

以Vue项目为例，建立Vite+Vue项目并运行只需四步：

```
yarn create vite-app <project-name>
cd <project-name>
yarn
yarn dev
```

Hello Vue 3.0 + Vite

count is: 0

Edit components/HelloWorld.vue to test hot module replacement.

Vite做了什么?

Vite的工作就是启用了一个koa服务。通过劫持浏览器的请求，在后端将所需文件进行分解整合，再返回给浏览器渲染页面，避免了Webpack打包的步骤，加快运行环境开发速度：

index.html > ...

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <link rel="icon" href="/favicon.ico" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Vite App</title>
8   </head>
9   <body>
10    <div id="app"></div>
11    <script type="module" src="/src/main.js"></script>
12  </body>
13  </html>
```

上图是入口文件index.html。在index页面中，当声明一个script标签为module时，浏览器会发起一个GET http://localhost:3000/src/main.js 请求main.js文件：

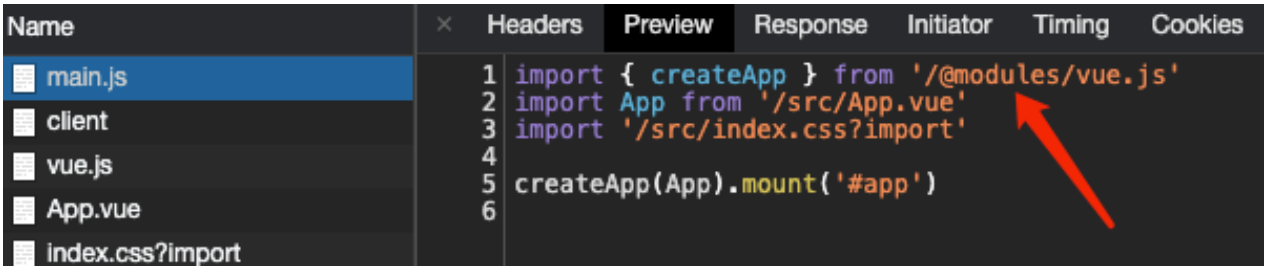
src > JS main.js

```
1   import { createApp } from 'vue'
2   import App from './App.vue'
3   import './index.css'
4
5   createApp(App).mount('#app')
```

开发环境的main.js文件

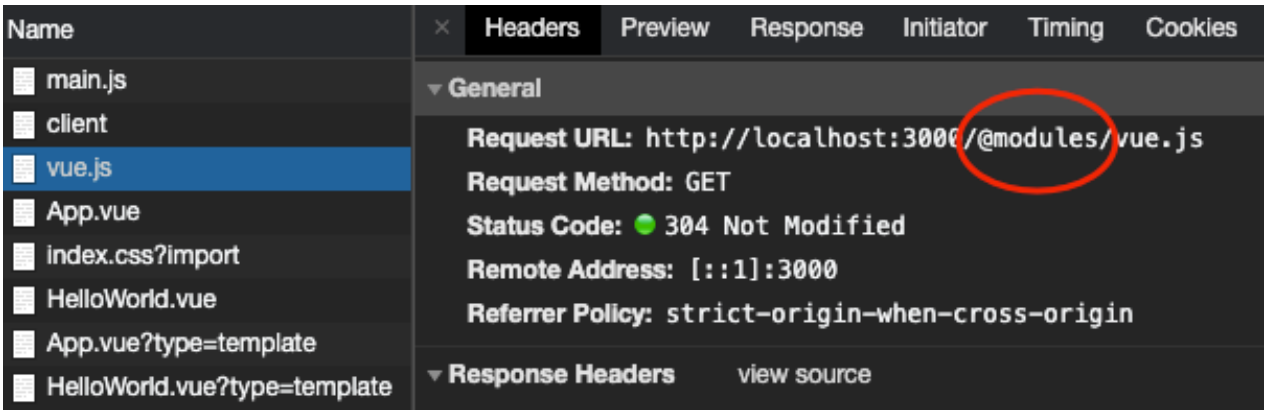
此时浏览器检测到含有import引入的包，又会发起GET请求来获取内容文件vue.js和App.vue。Vite主要做的工作是：

1.重写引入模块并拦截返回：



重写后的main.js文件

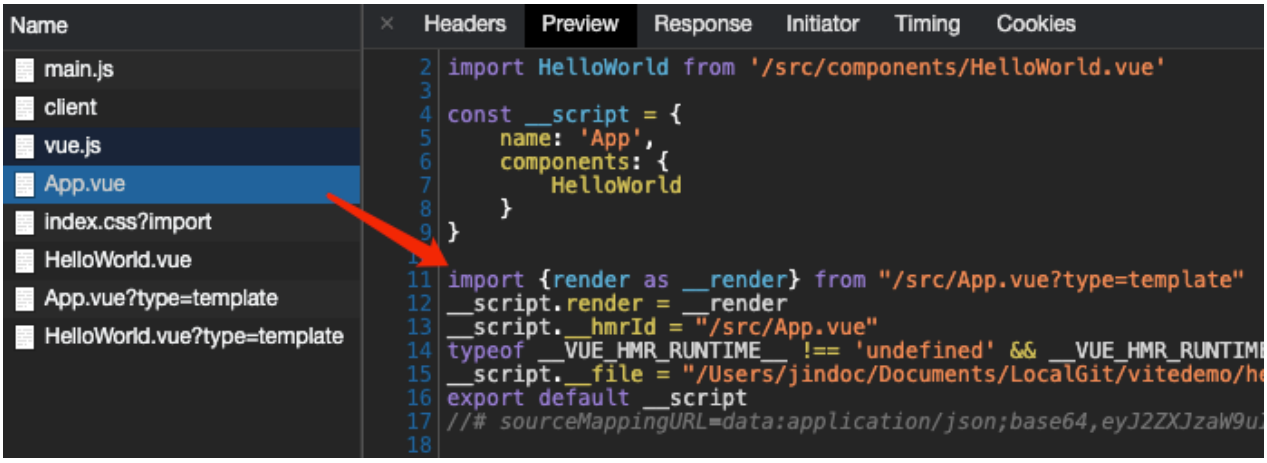
上图是浏览器开发者模式中看到的main.js文件。Vite将引入模块路径前加入了'/@modules/'，在浏览器发送请求时拦截含有'/@modules/'的请求，并从node_modules中引入相应的模块。



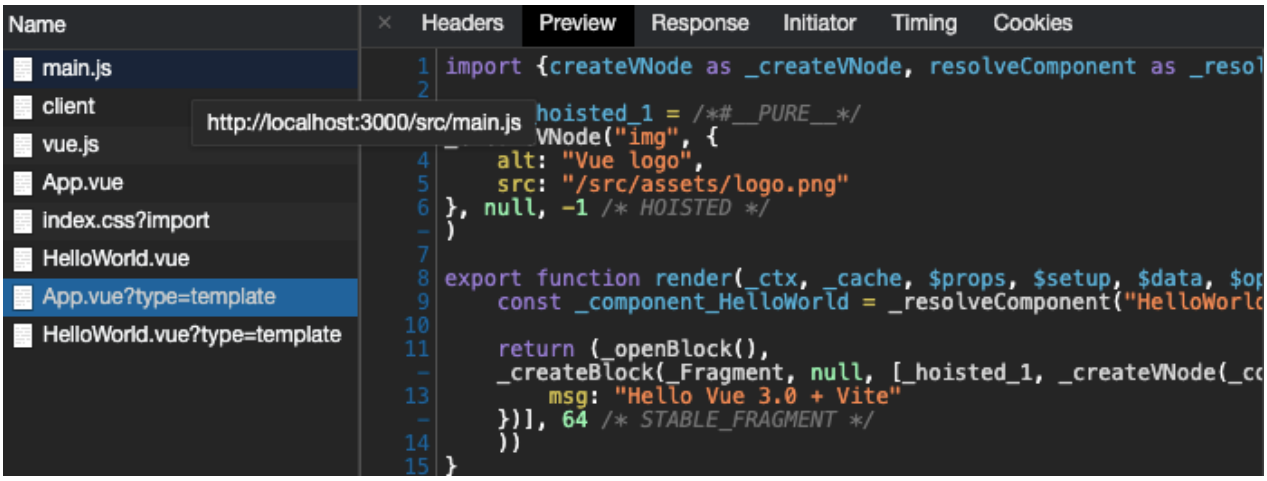
引入模块地址路径

2.解析.vue文件

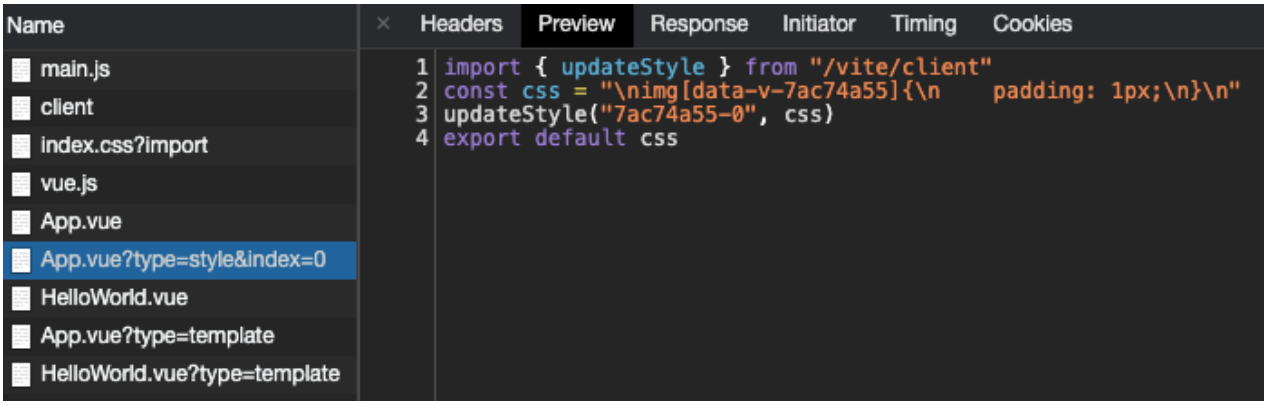
当koa中间件检测到请求是Vue Template，则在请求后添加?type=template参数，Vite将template解析成render函数返回给浏览器，而对于CSS用?type=style参数。



App.vue解析的JS



App.vue解析的Template

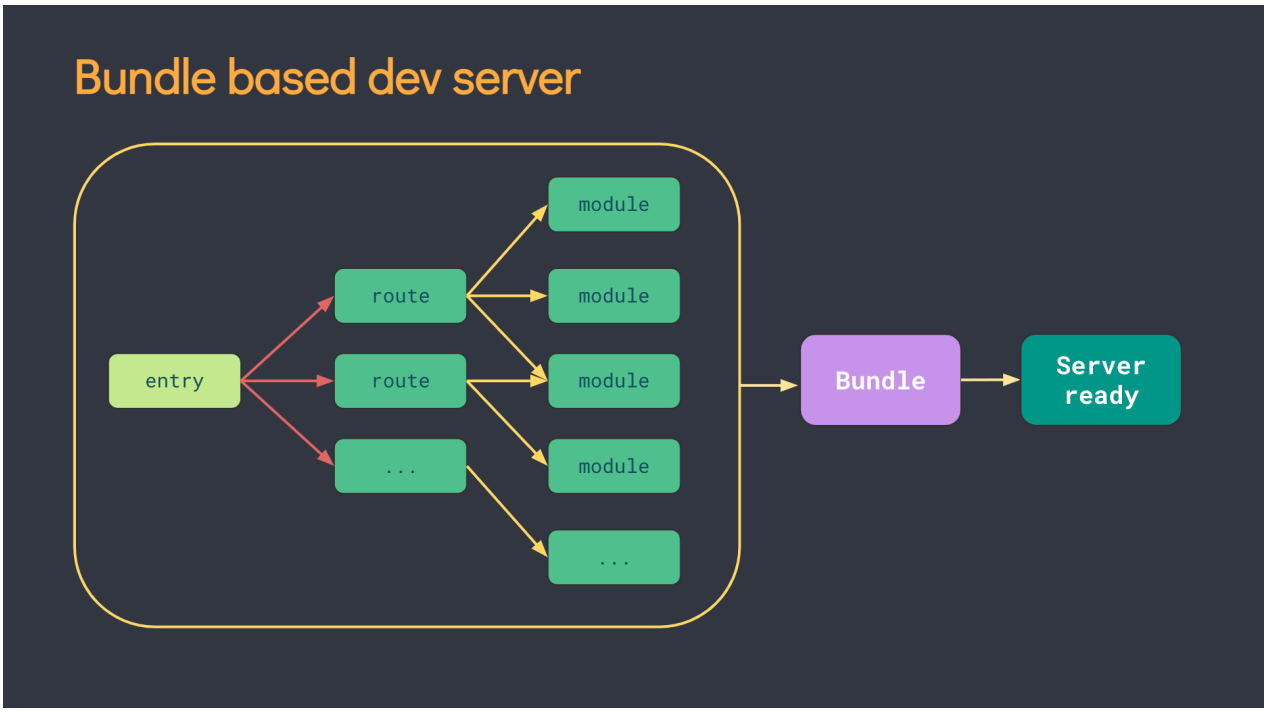


App.vue解析的CSS

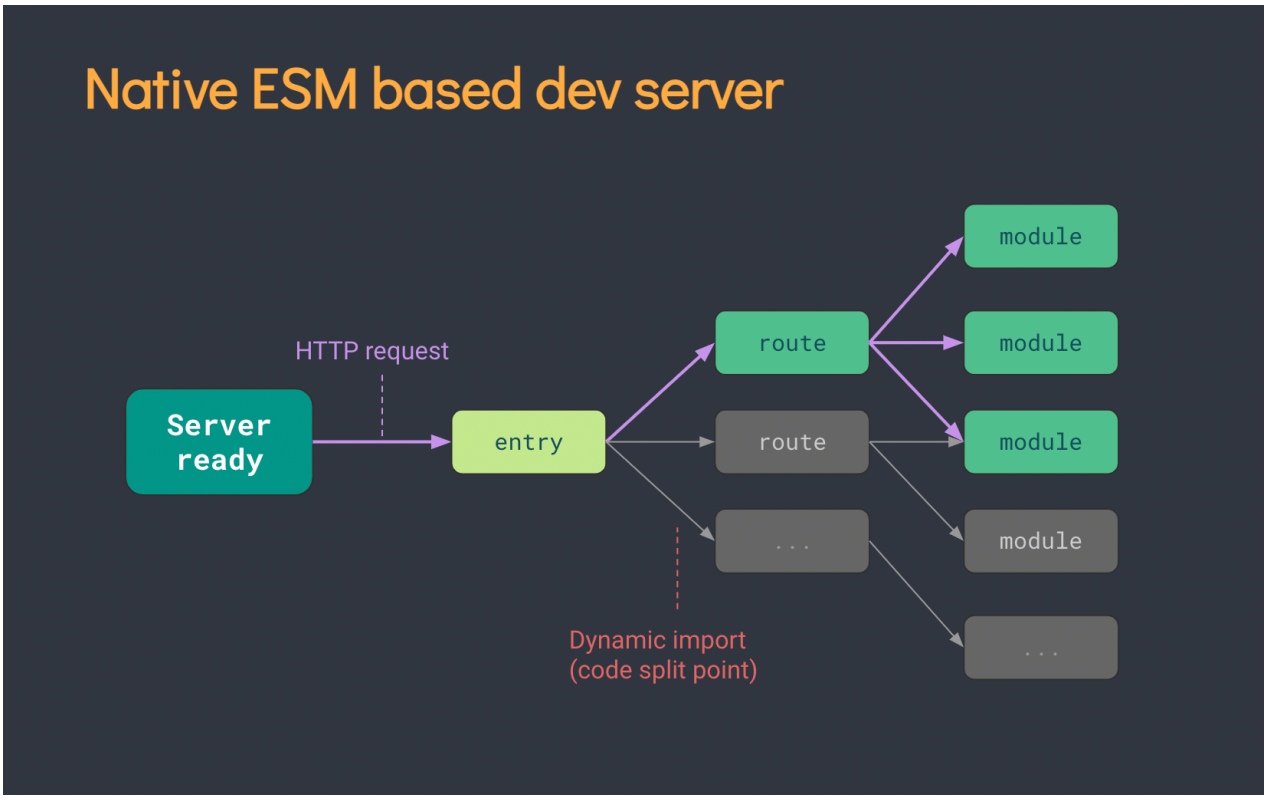
和JS、Template不同，CSS调用了client中的updateStyle，实现样式的热更新。

和webpack的不同

引用官方文档中的图来说明：



基于打包方式的开发服务器



基于EMS的开发服务器

在打包方式中(webpack为例)需要同时加载后才能展示单个路由页面。而Vite可以在浏览器的请求时按需编译和提供文件，仅需要当前路由页面使用的模块。同时Vite将每个文件都通过HTTP标头来进行缓存(若禁用浏览器缓存，则使用Vite的内存缓存)，页面更新时通过加入时间戳来重新请求文件。

总结

篇幅所限，本次仅大家介绍了Vite的基本原理。如果感兴趣，欢迎批评指正，与我们一起讨论。

参考

Vite官网：https://vitejs.dev/