第六课 hash 与 history 模式

学习目录

- **hash 与 history 模式区别**

一．hash 与 history 模式区别

hash 这个概念我们之前在说 Html 课程的时候也有说到过，是一个#符号来表示的，在 html 中我们用#符号来做页面定位。

vue 路由模式里的 hash 主要是在 url 中出现的，由于我们是单页面应用，所以 hash 模式下，我们的 http 请求只会给后端#符号前面的 url，因此不会影响到后端的配置，同时我们在改变路由 url 的时候不会刷新页面。因此 hash 模式的使用比较容易，但是颜值不太美丽。如果我们在不考虑 url 规范并且不需要后端人员配合的情况下使用 hash 模式肯定是没有问题的。

假如要考虑 url 规范，又比较在意 url 颜值，那么大家也可以考虑使用 history 模式，history 模式其实就是 vue 基于比较新的 HTML5 History Interface 中的 api 来封装的模式，这种模式最直观的感受是在 url 中没有#符号，符合 url 规范。

History 模式中使用了 pushState() 和 replaceState() 方法，他们可以直接作用到浏览器的历史记录栈，同时使用了 popState 事件监听状态变更。

虽然这种模式颜值美丽，符合 url 规范，但是需要后端人员配合，主要是因为我们通过 url 向后端发送请求，或者刷新页面的时候，可能会有 404 等错误提示。

比如说我们访问一些文章详情页面需要在 url 里使用动态路径参数，在后端人员不做路由匹配处理的情况下会返回一个 404 的错误。

假如请求的路由 url 是 http://192.168.3.196/article/id ,这里的 url 中有一个动态 id，如果后端缺少对 /article/id 的路由处理，将返回 404 错误，因此需要后端匹配对应好这个 url 才可以。

另外除了后端人员匹配好对应 url 之前，前端人员也可以增加一个全局 path 匹配，当发现 url 匹配不了设置好的路由 path 路径时，进入到一个 404 页面发出警告。

```
const router = new VueRouter({
  mode: 'history',
  routes: [
    { path: '*', component: NotFoundComponent }
  ]
})
```

总结一下，如果不在乎颜值和 url 规范，大家可以考虑使用 hash 模式，因为这个模式比较简单 ,后端人员也不用修改路由。如果考虑到颜值与 url 规范 ,那么可以考虑使用 history 模式。

谢谢观看！如果觉得课程还不错的话，记得给个好评！

我是星星课堂老师：周小周