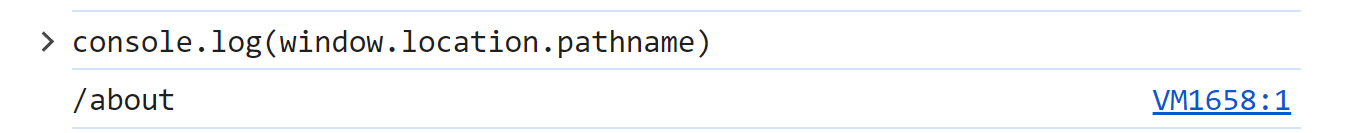
前言

我们在使用vue开发单页面应用的时候，需要使用动态组件和vue路由来完成页面内容的更新。你可能会问了，既然有了vue动态组件，为什么还要有vue路由呢？vue动态组件和vue路由之间的功能不是重叠了吗？

事实上vue路由的本质就是动态组件，它的目的是“根据浏览器地址的变化在页面上渲染对应的组件内容。”这句话很重要。使用它的好处是：不会导致页面重载从而防止页面刷新后用户已经输入的表单数据丢失、且支持地址历史记录的前进后退。

Vue路由的实现原理

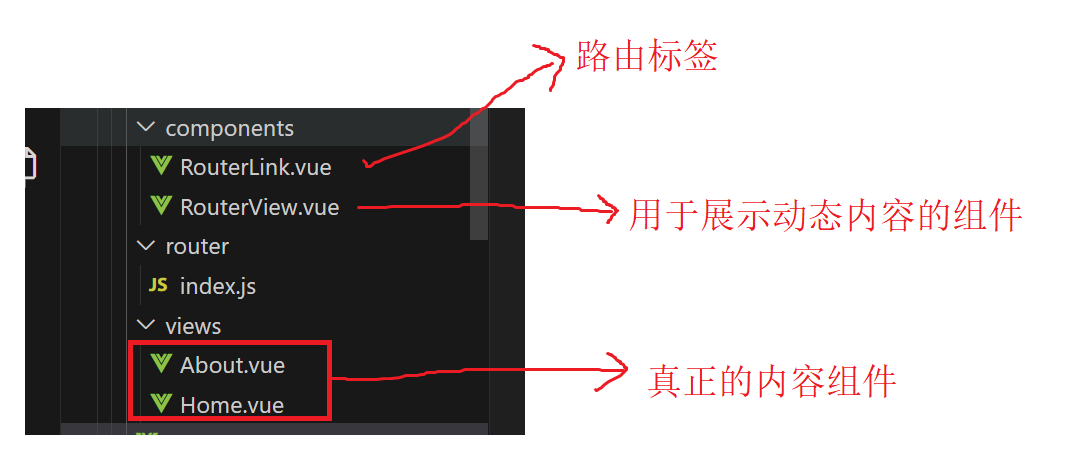
本质上它是通过使用vue的响应式数据，监听路径变量（这个路径变量是通过window全局对象获取的）的动态变化的同时，阻止a标签的默认行为来实现vue前端路由的。



具体的实现逻辑

我们需要定义一个用于展示动态组件内容的组件，同时需要一个组件用于展示和控制应该显示那个组件的选项卡组件(在vue3中就是router-link标签，这里我们也弄一个一摸一样的)。

紧接着就是用于动态展示的内容组件了。



在正式介绍如何具体是如何实现vue路由之前，你需要了解Vue的is 属性（attribute）和vue虚拟dom的基础知识。

我们先来看一下vue3官网动态组件对is属性的介绍：



看了上图你可能会有点懵，为什么可以是一个组件对象，或者原生html元素名呢？

哈哈哈，这里我简单的介绍一下vue虚拟dom的实现原理。这里你又需要了解一点点vue的基础知识（埋的坑有点多，没关系读完了，相信你就会对vue路由有更深刻的了解）：

1. 什么是虚拟dom
2. 什么是\*.vue文件。

什么是虚拟dom

我们知道在HTML页面中所有DOM元素都是一个object(“{}”)在这个object上有许多许多属性，用于标识具体是什么元素啦，这个dom元素中的内容啦，子元素啦，父元素啦，等等等等属性，这个内容是相当庞大的。

例如下图我们打印了一个真是的dom对象，我们来观察一下这个真实的dom对象上的属性。



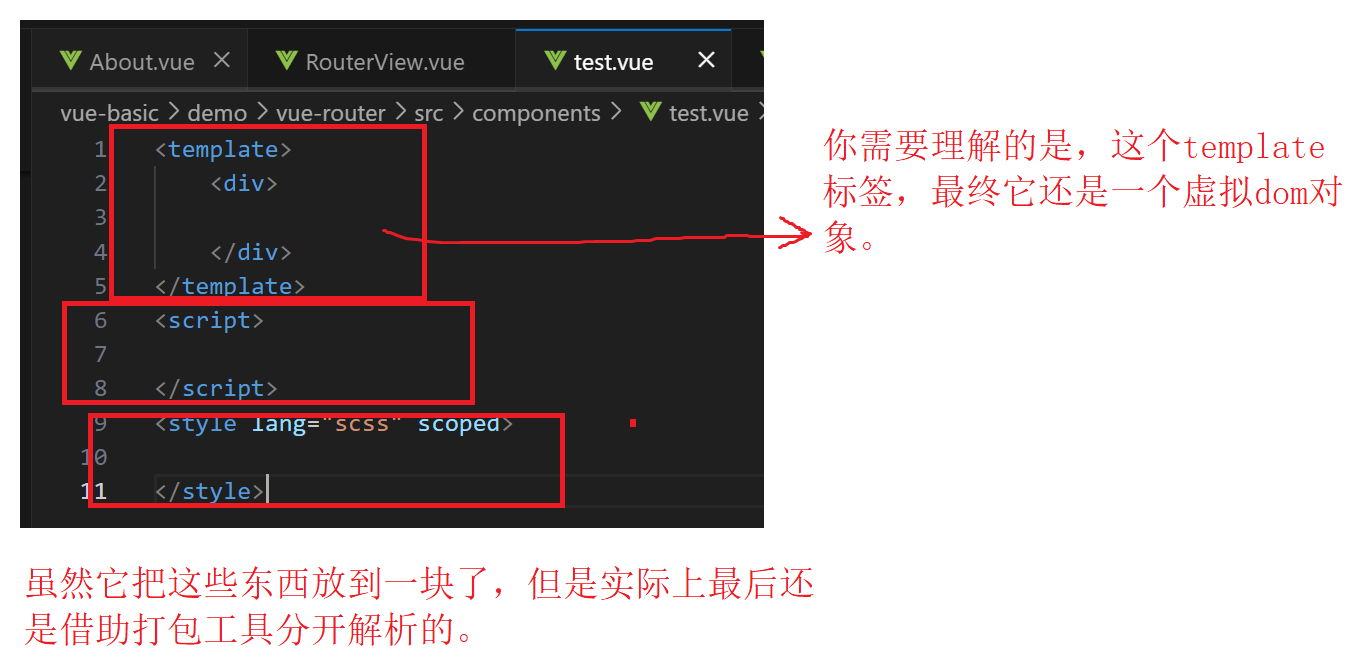
大概有个两三百个属性吧，但是在我们真正开发或者说操作dom元素的时候，很多很多属性我们是用不到的。我们知道操作属性这么多的对象的开销是非常大的（具体为什么开销大，需要你了解一下字典这种数据结构，JS,C++里的对象都是字典这种数据结构）。那何不把dom元素对象上的关键信息抽取出来，例如元素类型、内容、以及一个用于标记对这个元素做什么操作的标签属性，最终再用Js给我们提供的操纵html元素的api来完成（注意：使用虚拟dom的好处还不只这些，它还有助于提升diff算法的时间复杂度等）。

什么是\*.vue文件

在学习vue的过程中，相信你一定使用过在vue组件实例上的template属性上创建vue模板内容的方式。比如说下图种的用法：  


你觉得自己是使用写原生html元素的写法写这些内容，但是实际上这些内容都要经过vue的模板预编译器编译的，编译的结果实际上也是一个虚拟dom对象。最终再交由vue的render()方法去向页面上渲染的。

说了这么多，终于可以介绍什么是\*.vue文件了。。。。。



解答为什么vue中is属性的值可以是组件对象或原生HTML元素名。

1. is的值如果是一个“span”字符串，他就使用render()方法帮你创建一个span标签（这个东西很简单，就是利用document.createElement()方法就行）。
2. 如果是一个虚拟dom对象（也就是组件对象，也叫组件实例），那它还是利用render()方法去解析这个对象（object,“{}”）然后用Js提供的api去创建并渲染dom元素的。

Vue本质上没有什么特别复杂的东西，就仿佛一个机器，揭开来看，其实里面并没有什么很复杂的零件。

真正的实现vue路由

首先，我们需要能够根据浏览器地址栏的地址变化，动态的提供组件的js文件

这个js文件要提供：

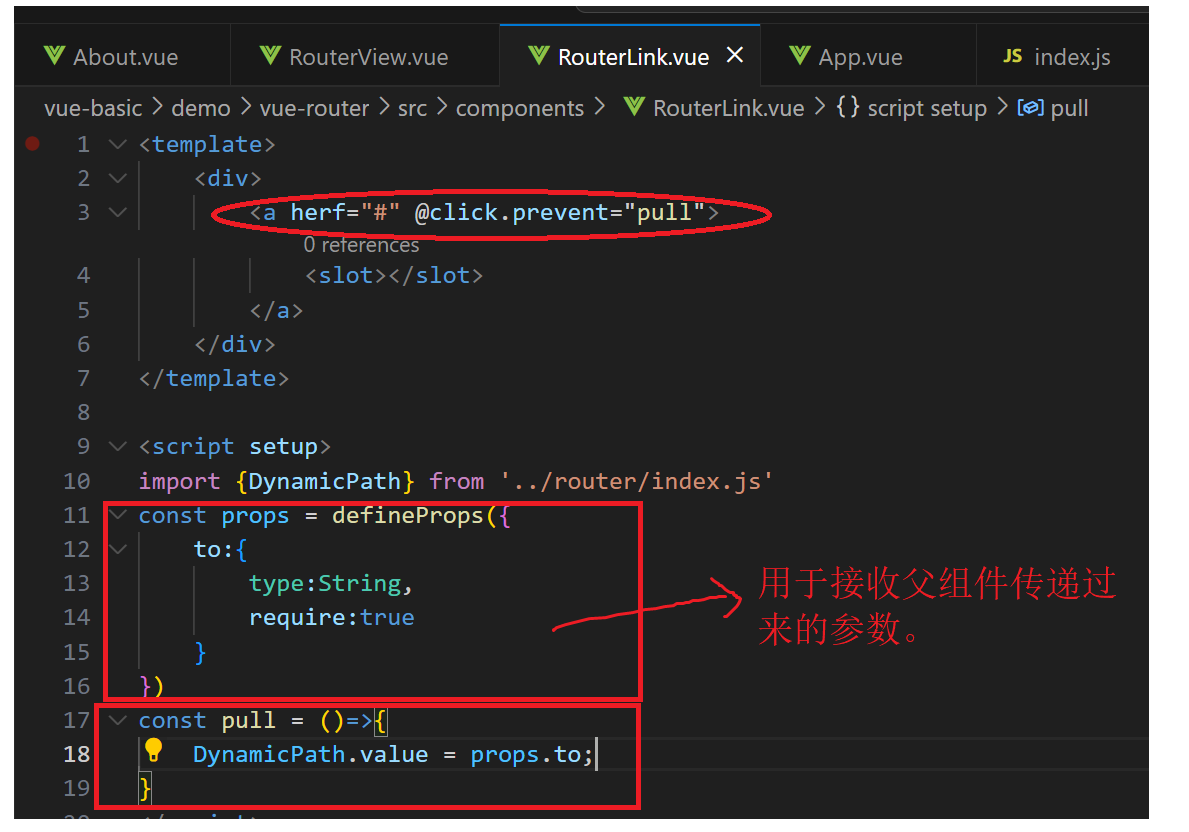
1. 能根据浏览器的地址栏的地址的不同而动态改变的数据。
2. 要提供与对应地址相匹配的组件对象。



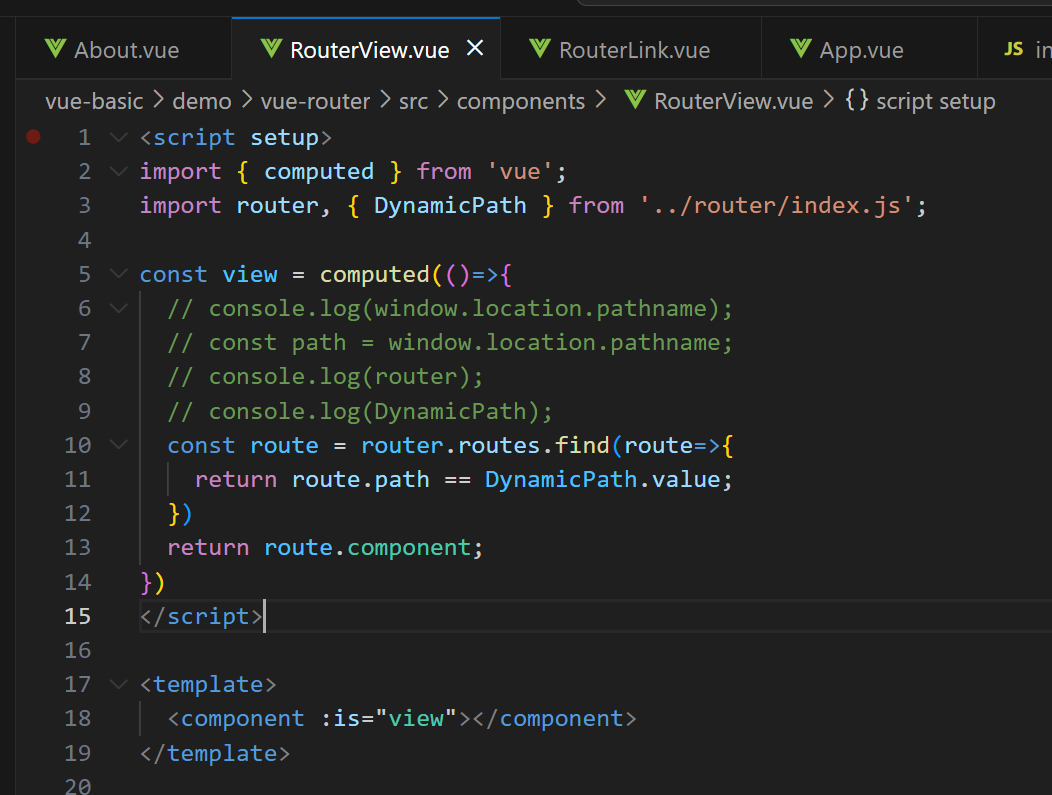
写到这其实我们的核心逻辑已经讲完，只是还没有在具体的组件中实现它。我们先来看看

RouterLink组件的作用与功能。

1. 我们需要在RouterLink组件中完成点击a标签后切换路径的操作（再重复一遍这个路径是一个响应式数据，可供vue监听的），最终要展示组件内容的地方就是根据这个来切换不同的组件的。
2. 要接收传过来路径。

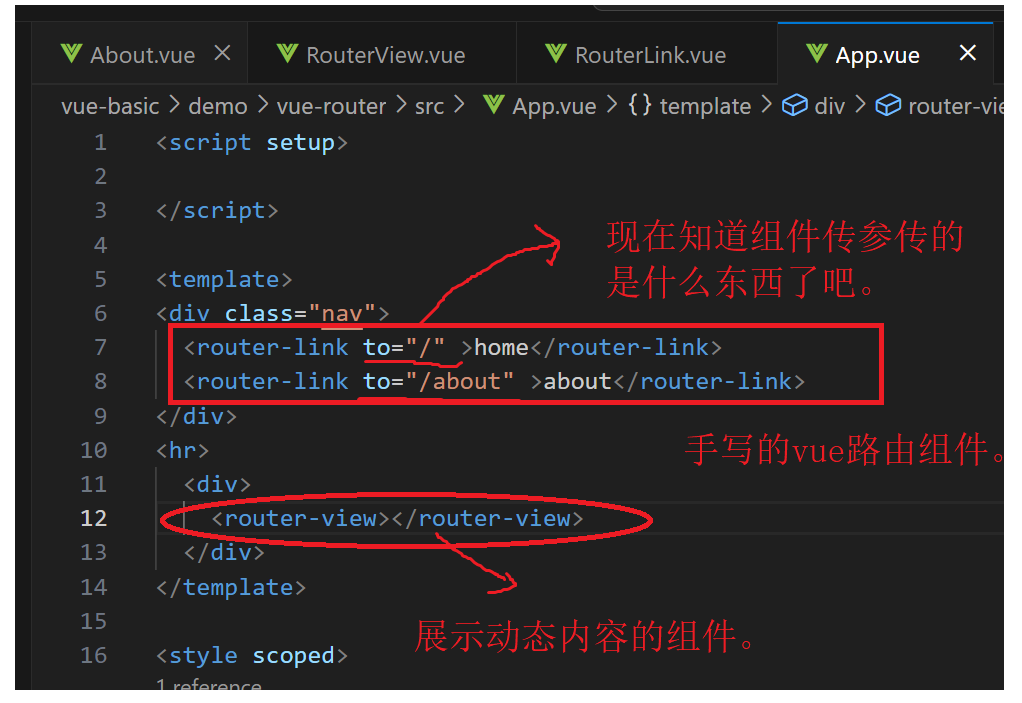


我们再来看看用于展示动态内容组件的组件



没什么很深的逻辑，这个view也是一个计算属性，它能够根据DynamicPath的变化返回不同的组件对象，”:is”就是读取这个组件对象来实现组件的切换。

最后我们再来看看如何使用RouterLink组件。



到这整个vue路由的原理我们已经介绍完了，这个小demo我会上传到github中，感兴趣可以拉取到本地测试运行。