# vue-router基础

一, 动态路由匹配

需求：我们经常需要把某种模式匹配到的所有路由，全都映射到同个组件。

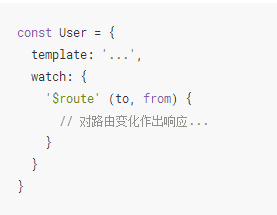


参数：当匹配到一个路由时，参数值会被设置到 this.$route.params

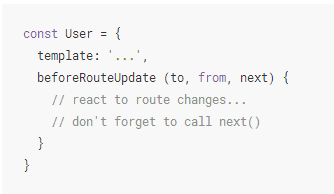
提醒：当使用路由参数时，例如从 /user/foo 导航到 user/bar，原来的组件实例会被复用，这也意味着组件的生命周期钩子不会再被调用。

如果想在复用时，对路由参数的变化做出响应的话，

1，可以通过watch(监测) $route对象



2，或者使用 beforeRouteUpdate 守卫

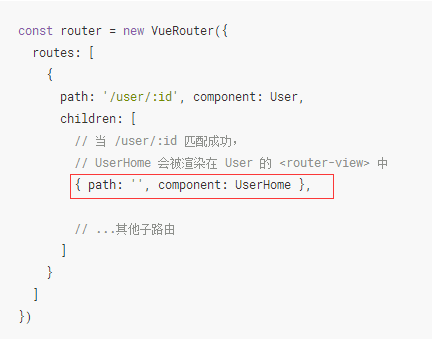


二，嵌套路由



注意：以 / 开头的嵌套路径会被当作根路径。 这让你充分的使用嵌套组件而无须设置嵌套的路径。

需求--当没有匹配到合适的子路由时，匹配默认组件：

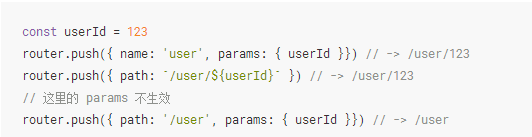


三，编程式的导航

1，$router.push跳转：



注意：如果提供了 path，params 会被忽略。



router.replace：跟 router.push 很像，唯一的不同就是，它不会向 history 添加新记录，而是跟它的方法名一样 —— 替换掉当前的 history 记录。



提示：router.push、 router.replace 和 router.go 跟 [**window.history.pushState、window.history.replaceState 和 window.history.go**](https://developer.mozilla.org/en-US/docs/Web/API/History)好像， 实际上它们确实是效仿 window.historyAPI 的。

四，命名路由



5，命名视图

需求：有时候想同时（同级）展示多个视图，而不是嵌套展示。



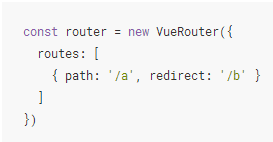
6，重定向 和 别名

重定向：当匹配/a时，重定向到/b；

别名：当匹配到/a时，还是/a；当匹配到/b时，路径还是/b，但是实际匹配的是/a；

重定向也是通过 routes 配置来完成，

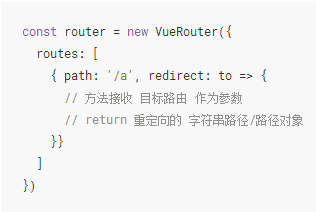
下面例子是从 /a 重定向到 /b(字符串)：



重定向的目标也可以是一个命名的路由：



甚至是一个方法，动态返回重定向目标：



上面别名对应的路由配置



7，向路由组件传递props

省略...

8，HTML5 History 模式

因为我们的应用是个单页客户端应用，如果后台没有正确的配置，当用户在浏览器直接访问 http://oursite.com/user/id 就会返回 404，这就不好看了----所以需要后台配置支持

后台配置所要达到的效果：如果 URL 匹配不到任何静态资源，则应该返回同一个index.html 页面，这个页面就是你 app 依赖的页面----匹配不到资源，就返回到 主页。

有时为了避免[对于服务器就再返回 404 错误页面，所有路径都会返回 index.html 文件]这种情况，你应该在 Vue 应用里面覆盖所有的路由情况，然后在给出一个 404 页面。

