vue-router的理解2

路由元信息

滚动行为

数据获取

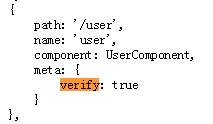
组件懒加载

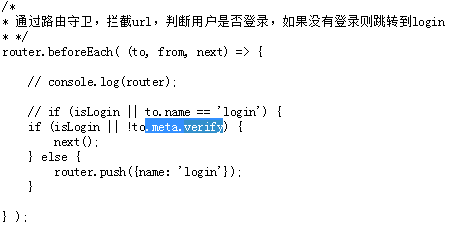
路由其他API

路由元信息

通过meta选项，我们可以给路由设置自定义的数据

应用：比如点击进入个人中心，会去判断用户有没有登录，进入其他页面不需要验证。如果有很多个个人中心，要一个个去写判断条件，比较麻烦，所以，就可以在路由配置参数里面设置一个meta属性，写判断是只要判断有没有这个属性，有这个属性，就跳转到登录页面





**滚动行为**

路由跳转的时候设置页面的滚动条位置

为什么存在：切换组件，滚动条应该在更新的组件的最上面，但是vue页面是不会这么处理的，因为vue写的单页面。

设置：在路由的配置参数中添加这个属性

实现：每次导航发生的时候，控制页面滚动到顶部

scrollBehavior (to, from, savedPosition) {

return { x: 0, y: 0 }

}

第三个参数：savedPosition

记录离开导航的位置

当按(通过浏览器的 前进/后退 按钮触发) 时才可用。

数据获取

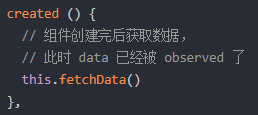
路由完成以后加载数据

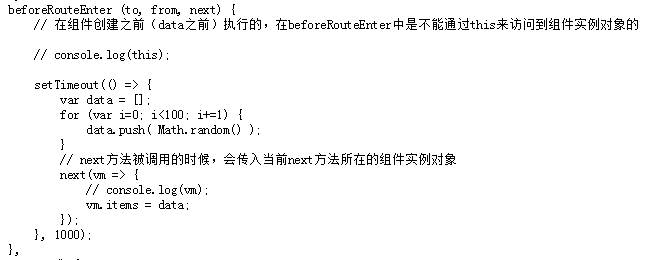
组件的生命周期钩子函数中处理数据——create

加载完数据以后跳转路由

组件的路由守卫函数中处理数据——beforeRouteUpdate

注意：在组件创建之前（data之前）执行的，在beforeRouteEnter中是不能通过this来访问到组件实例对象的。需要在next第一个参数中找，第一个参数就是这个实例对象





组件懒加载

把组件打包成独立的文件，按需加载组件文件

const Foo = () => import(/\* webpackChunkName: "group-foo" \*/ './Foo.vue')

打包 >npm run build

打包后的文件：

mainfest文件清单，包含哪些子文件，对app,vendor的说明，作用是什么

vendor打包后的vue的核心文件

app写的js代码

作用：

分开打包，体积小

多入口文件,vue核心文件复用性高

路由其他API

<router-link>

to

replace

append——以当前路径为根路径，和路由的children类似。平常还是用children

**tag**——替换当前的标签

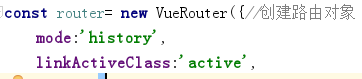
active-class——自动添加class名字router-link-active，

**exact-active-class**——精确匹配路径，添加class名字router-link-active，

exact——精确匹配to里面的路径

event——默认是click，现在可以自定义

两种方法设置active高亮状态：



**或：**

clipboard.png

Router配置参数

**mode**——"hash" | "history" | "abstract"

**base**——根路径

linkActiveClass——类似active-class

linkExactActiveClass——类似exact-active-class

scrollBehavior（to,from,savedPosition）

parseQuery / stringifyQuery——自定义解析query和params