# 路由

## API参考

<https://router.vuejs.org/zh/api/>

## 关于hash

<https://www.cnblogs.com/joyho/articles/4430148.html>

## 基本示例

首先引入vue.js和vue-router.js

<script src="https://cdn.bootcdn.net/ajax/libs/vue/2.6.12/vue.js"></script>

<script src="https://cdn.bootcdn.net/ajax/libs/vue-router/3.4.8/vue-router.min.js"></script>

<**div id="box"**>  
 <**router-link to="/login"**>登录</**router-link**>  
 <**router-link to="/register"**>注册</**router-link**>  
  
 *<!-- <router-view> 是vue-router提供的元素，可以认为是一个占位符，  
 将来，路由规则匹配到的组件，就会展示在<router-view>处   
 就是使用router-view指定组件渲染的位置-->* <**router-view**></**router-view**>  
</**div**>  
<**script**>  
 *// 定义组件的模板对象：  
 // 可以从其他文件 import 进来  
 // 可以是通过Vue.extend创建的，* **let *Login*** = Vue.extend({  
 **template**: **'<div>我要登录</div>'** });  
  
 *// 可以是一个组件配置对象  
 // 上面的写法等价于下面这种写法:  
 // const Login = { template: '<div>我要登录</div>' }* **let *Register*** = Vue.extend({  
 **template**: **'<div>我要注册</div>'** });  
  
 *// 注意：组件名称仅限于在HTML中通过 <register></register>调用;   
 // 不可以作为component的属性值  
 // Vue.component('register', {  
 // template: '<div>我要注册</div>'  
 // })  
  
 // 创建一个路由对象，当导入vue-router包之后，在window全局对象中，  
 // 就有了一个路由的构造函数，叫做VueRouter  
 // 在new路由对象的时候，可以为构造函数传递一个配置对象* **const *router*** = **new** VueRouter({  
 **routes**: [ *// routers 表示路由匹配规则  
 // 每个路由规则，都是一个对象，这个对象有两个必须的属性：  
 // path: 表示监听哪个路由链接地址； component:表示当前路由地址path所对应的组件  
 // 注意：component的属性值，必须是一个组件的模板对象，不能是组件名称  
 // 当component的属性值是一个【组件配置对象】的时候，  
 // vue-router内部会自动调用Vue.extend构造器* { **path**: **'/login'**, **component**: ***Login*** },  
 { **path**: **'/register'**, **component**: ***Register*** },  
 ]  
 });  
  
  
 **new** Vue({  
 **el**: **"#box"**,  
 ***router***, *//将路由对象，注册到vm实例上，用来监听URL地址的变化，然后展示对应的组件* })  
</**script**>

## 使用超链接实现路由跳转

由于vue默认是通过hash模式控制路由的，所以超链接前面需要加#

<**a href="#/home"**>首页</**a**>  
<**a href="#/about"**>关于我们</**a**>  
<**a href="#/news"**>新闻</**a**>

## 使用router-link

使用a标签，会造成页面的重新刷新，这就不是单页面应用了，而是一个多页应用。当然可以取消默认行为，但是每次都要手动去掉默认行为很麻烦。

我们有更好的方法，使用router-link

<**nav**>  
 <**router-link to="/home"**>首页</**router-link**>  
 <**router-link to="/about"**>关于我们</**router-link**>  
 <**router-link to="/news"**>新闻</**router-link**>  
</**nav**>  
<**router-view**></**router-view**>

这种方式还有一个好处：把history模式去掉，它依然可以正常使用。

## router-link的to地址可以绑定到data数据

<router-link :to="index">首页</router-link>

data() {

return {

index: 'home'

}

},

## router-link的to地址可以绑定为一个对象

<router-link :to="{path:'/about'}">关于我们</router-link>

## 修改router-link生成的默认标签

router-link默认生成的是超链接a标签，可以通过tag属性修改：

<router-link to="/news" tag="div">新闻</router-link>

通过控制台看到默认的a标签已经变成div标签了，跳转依然正常

使用li标签加图标示例：

<**router-link to="/product" tag="li"**>  
 <**i class="iconfont icontiyu"**></**i**> <**span**>产品</**span**>  
</**router-link**>



下面这样写，渲染时会自动把跳转地址添加到a上面，因此a是不需要添加href属性的：

<**router-link to="/product" tag="li"**>  
 <**a**>产品</**a**>  
</**router-link**>

## 通过router-link-active控制高亮样式

**a**.**router-link-active** {  
 **background-color**: **pink**;  
}

## 自定义高亮样式的class名称

在路由实例中增加选项linkActiveClass

**new** VueRouter({  
 **mode**: **'history'**,  
 **linkActiveClass**: **'is-active'**,  
 **routes**: [  
 { **path**: **'/home'**, **component**: Home },  
 { **path**: **'/about'**, **component**: About },  
 { **path**: **'/news'**, **component**: News }  
 ]  
});

**a**.**is-active** {  
 **background-color**: **yellow**;  
}

## 单独设置某一个标签的高亮样式

<router-link to="/news" active-class="on">新闻</router-link>

**a**.**on** {  
 **background-color**: **red**;  
}

## 在router-link标签上添加事件

默认是点击切换路由，可以通过添加事件切换路由

<**nav**>  
 <**router-link :to="index" event="mouseenter"**>首页</**router-link**>  
 <**router-link :to="{path:'/about'}" event="mouseenter"**>关于我们</**router-link**>  
 <**router-link to="/news" active-class="on" event="mouseenter"**>新闻</**router-link**>  
</**nav**>

## 添加404页面

在路由实例中添加：

{ path: '\*', component: NotFound }

在浏览器中输入一个不存在的路由：http://localhost:8080/abcd就会显示404页面

## 重定向redirect

访问根目录时，可以让它匹配home组件: { **path**: **'/'**, **component**: Home },  
但是不推荐这么做，因为访问home是显示home组件，访问根目录是也是显示home组件，分不清楚；因此推荐使用重定向去做：

**new** VueRouter({  
  
 **linkActiveClass**: **'is-active'**,  
 **routes**: [

{ **path**: **'/'**, **component**: Home },  
 { **path**: **'/home'**, **name**: **'HomePage'**, **component**: Home },  
 { **path**: **'/about'**, **name**: **'AboutPage'**, **component**: About },  
 { **path**: **'/news'**, **name**: **'NewsPage'**, **component**: News },  
 {  
 **path**: **'\*'**,  
 *// redirect: '/home' // 可以是个字符串  
 // redirect: {path: '/home'} // 可以是个对象  
 // redirect: {name: 'HomePage'} // 可以是name,name值就是每个路由中的name值* redirect: (toobj) => { *// 可以是一个函数，动态设置重定向目标  
 // toobj => 目标路由对象，就是访问的路径的路由信息* ***console***.log(toobj);  
  
 *// return '/home'* **if** (toobj.**path** == **"/123"**) {  
 **return '/home'** } **else if**(toobj.**path** == **"/456"**) {  
 **return** {**path**: **'/about'**}  
 } **else** {  
 **return** {**name**: **'NewsPage'**}  
 }  
  
 }  
 }  
 ]  
});

说明：在浏览器地址栏输入http://localhost:8080/123就会显示home页面

## 别名

通过添加alias属性设置别名，浏览器地址栏输入http://localhost:8080/index就会显示home页面

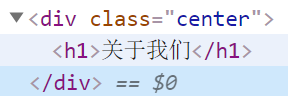
{ path: '/home', name: 'HomePage', component: Home, alias: '/index' },

## router-view上添加class

在router-view标签上添加的class会自动渲染到每个组件的根标签上面

<router-view class="center"></router-view>

根组件的div上面多了一个class



## 根路径的设置

把根目录映射到根组件，访问其他路径时 ，首页也会高亮，解决办法：添加exact精确匹配

<router-link to="/" exact>首页</router-link>

{ path: '/', component: Home },

## 子路由

在news路由配置中通过children添加子路由配置

{  
 path: **'/news'**, ***name***: **'NewsPage'**, component: News,  
 children: [  
 { **path**: **'guoji'**, **component**: GuoJi },  
 { **path**: **'guonei'**, **component**: GuoNei },  
 { **path**: **'redian'**, **component**: ReDian },  
]  
}

在news子组件中添加路由router-link跳转  
添加router-view告诉路由组件渲染的位置

<**h1**>新闻中心</**h1**>  
<**nav**>  
 <**router-link to="/news/guoji"**>国际新闻</**router-link**>  
 <**router-link to="/news/guonei"**>国内新闻</**router-link**>  
 <**router-link to="/news/redian"**>热点新闻</**router-link**>  
</**nav**>  
  
<**router-view**></**router-view**>

## 设置默认子路由

只需要把某个子路由的path设置为空即可：

{  
 path: **'/news'**, ***name***: **'NewsPage'**, component: News,  
 children: [  
 { **path**: **''**, **component**: GuoJi },  
 { **path**: **'guonei'**, **component**: GuoNei },  
 { **path**: **'redian'**, **component**: ReDian },  
]  
},

访问/news路由时会默认显示子组件 GuoJi ,但是此时国际路由不会高亮，解决方法：

<**router-link to="/news/guoji"**>国际新闻</**router-link**>

改成：

<**router-link to="/news" exact**>国际新闻</**router-link**>

注意：此时控制台会有个警告：如果设置了子路由，就不要在父路由里面添加name选项。解决方法：把name属性添加到默认子路由中，子路由的其他映射组件也都可以添加name属性：

{  
 path: **'/news'**, component: News,  
 children: [  
 { **path**: **''**, **name**: **'NewsPage'**, **component**: GuoJi },  
 { **path**: **'guonei'**, **name**: **'GuoNeiPage'**, **component**: GuoNei },  
 { **path**: **'redian'**, **name**: **'ReDianPage'**, **component**: ReDian },  
]  
},

## 子路由也可以绑定到name

<**nav**>  
 <**router-link :to="{name: 'NewsPage'}" exact**>国际新闻</**router-link**>  
 <**router-link :to="{name: 'GuoNeiPage'}"** >国内新闻</**router-link**>  
 <**router-link :to="{name: 'ReDianPage'}"** >热点新闻</**router-link**>  
</**nav**>

## 子路由相对父级路径访问

子路由path不加 / , 那么浏览器访问地址就是：http://localhost:8080/news/guonei

如果给子路由path加上/ ，那么在浏览器访问地址默认就是相对于news, 就可以省略父级路径：  
http://localhost:8080/guonei

{  
 path: **'/news'**, component: News,  
 children: [  
 { **path**: **''**, **name**: **'NewsPage'**, **component**: GuoJi },  
 { **path**: **'/guonei'**, **name**: **'GuoNeiPage'**, **component**: GuoNei },  
 { **path**: **'/redian'**, **name**: **'ReDianPage'**, **component**: ReDian },  
]  
},

## 动态路由

使用v-bind动态绑定路由地址，item.id来自于ajax请求的数据

<**nav**>  
 <**router-link :to="'/user/' + item.id" v-for="(item,index) in userlist" :key="index"**>  
 {{item.name}}  
 </**router-link**>  
</**nav**>

{ path: '/user/:userid', component: User },

## 路由对象

$router router实例对象  
$route 当前激活的路由信息对象，每个组件实例都会有  
beforeRouteEnter() 进入组件前钩子函数  
beforeRouteLeave() 离开组件前钩子函数

### $route路由信息对象

一个路由信息对象表示当前激活的路由的状态信息，每次成功的导航后都会产生一个新的对象

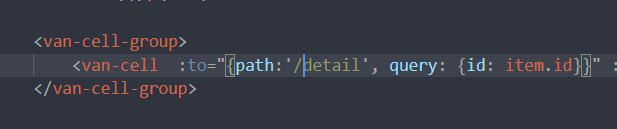
path 字符串，对应当前路由的路径  
params 对象，包含动态路由参数  
query 对象，URL查询参数  
hash 字符串，当前路由的hash值  
fullPath 字符串，URL包含查询参数和hash的完整路径  
matched 数组，包含当前路由的所有嵌套路径片段的路由记录  
name 字符串，当前路由的名称

## params和query

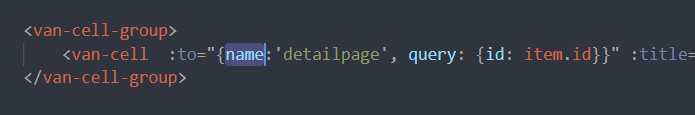
https://www.jb51.net/article/163764.htm

### query

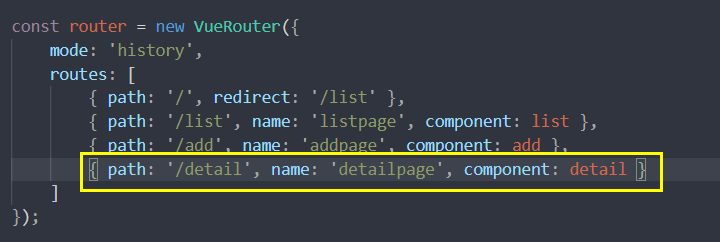
点击列表页跳转：



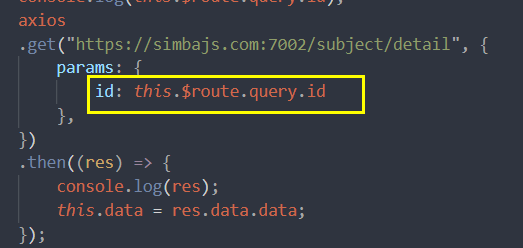
也可用name：



注意组件路由index.js中path格式：



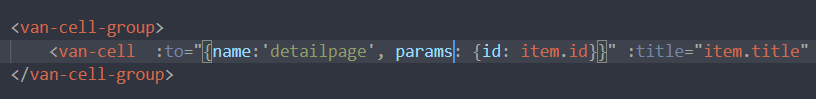
detail页面获取id



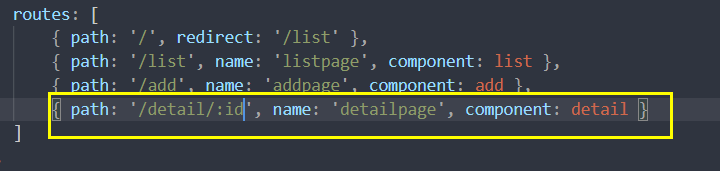
### params

只能用name不能用path，不然会直接无视掉params中的内容

列表页：



路由index.js中path：



detail页面：

