一:基本步骤

1:导入路由的包

2：在<script>标签中创建一个路由对象

```
var vueRouter = new vueRouter({

    // 每个路由规则，都是一个对象，这个规则对象，身上，有两个必须的属性：
    //  属性1 是 path， 表示监听 哪个路由链接地址；
    //  属性2 是 component， 表示，如果 路由是前面匹配到的 path ，则展示 component 属性对应的那个组件
    // 注意： component 的属性值，必须是一个 组件的模板对象， 不能是 组件的引用名称；
    routers:[

{ path: '/login', component: login },....

];
});
```

3:将路由对象与vue实例关联起来

```
var vm = new Vue({
    el: "#app",
    data: {},
    methods: {},
    components: {
        'register': register,
        'login': login
    },
    router:vueRouterObj
        //将路由规则对象，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

    // } //

});
```

4：在vue控制的区域内 加入下面的标签

```
<!-- 这是 vue-router 提供的元素，专门用来 当作占位符的，
    将来，路由规则，匹配到的组件，就会展示到这个 router-view 中去 -->
<!-- 所以： 我们可以把 router-view 认为是一个占位符 -->
<router-view></router-view>
```

二：router-link的使用

```
<router-link to="/login" tag="sapn">登陆</router-link>
<router-view to="/register">注册</router-view>
```

router-link 标签默认会渲染为一个 a 标签

to属性 ：表示 要跳转到的地址，双引号中 不用 加 #

tag属性：可修改 router-link 将要渲染的标签

三：重定向的使用

```
var routerObj = new VueRouter({

        routes: [
            { path: '/', redirect: '/login' }, // 这里的 redirect 和 Node 中的 redirect 完全是两码事
            { path: '/login', component: login },
            { path: '/register', component: register }
        ]
    })
```

如上例中：打开界面后，默认去 login组件的界面

四：选中的地方高亮显示，（或者额外添加样式）

方法1： style标签中在 .router-link-active 类中书写想要的样式

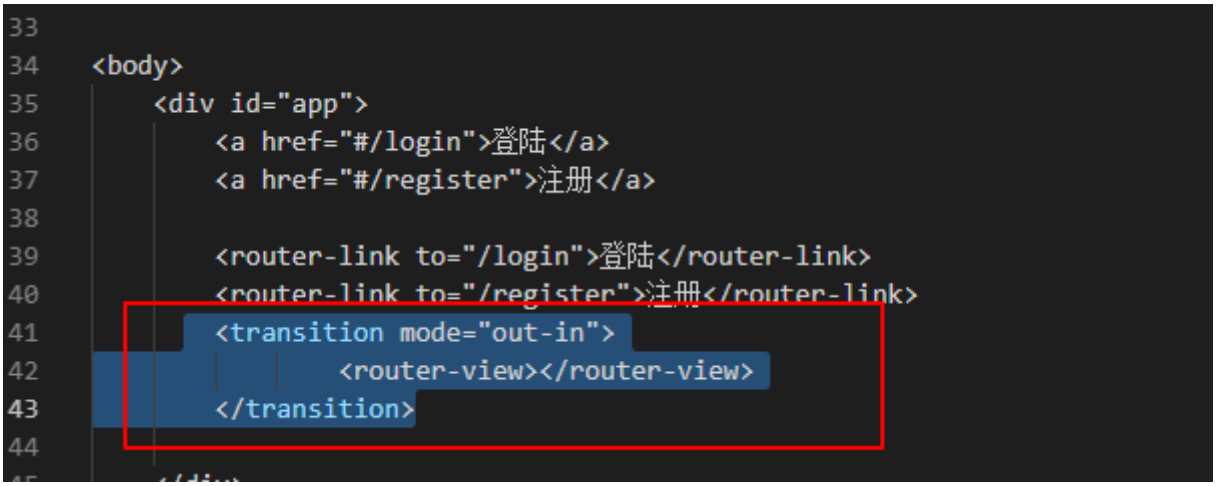
）

方法2：

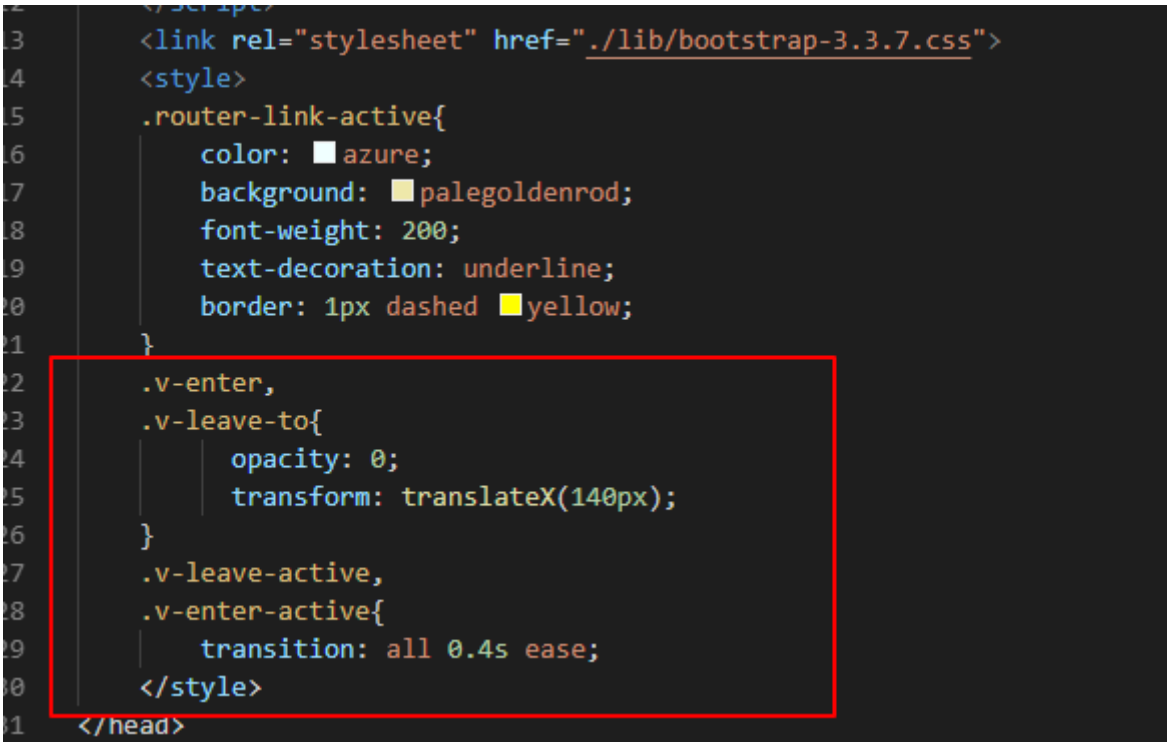
```
var routerObj = new VueRouter({
    注意：这块不是routers,没有r
        routes: [
            { path: '/', redirect: '/login' }, // 这里的 redirect 和 Node 中的 redirect 完全是两码事
            { path: '/login', component: login },
```

```
            { path: '/register', component: register }
        ],
//添加linkActiveClass 选项，值用引号引住，然后在style中为
.myactive的样式
linkActiveClass: 'myactive'


    })
```

五：为路由添加动画



```
<body>
    <div id="app">
        <a href="#/login">登陆</a>
        <a href="#/register">注册</a>

        <router-link to="/login">登陆</router-link>
        <router-link to="/register">注册</router-link>
        <transition mode="out-in">
            <router-view></router-view>
        </transition>
```



```
    <link rel="stylesheet" href="./lib/bootstrap-3.3.7.css">
    <style>
    .router-link-active{
        color: azure;
        background: palegoldenrod;
        font-weight: 200;
        text-decoration: underline;
        border: 1px dashed yellow;
    }
    .v-enter,
    .v-leave-to{
        opacity: 0;
        transform: translateX(140px);
    }
    .v-leave-active,
    .v-enter-active{
        transition: all 0.4s ease;
    </style>
</head>
```

六：在路由地址中定义参数：

方式一：

如下 tag 的意思是 将<router-link>渲染成 指定元素 如 span

使用 ？ &传参

```
 <router-link to="/login?id=10&name=zs" tag="span">登录</router-link>
```

以query的方式取值

```
<script>
  // 组件的模板对象
  var login = {
    template: '<h1>登录{{$route.query.id}}==========={{$route.query.name}}组件</h1>'
  // template: '<h1>登录{{$route.params.id}}==========={{$route.params.name}}组件</h1>'
}
```

方式二：

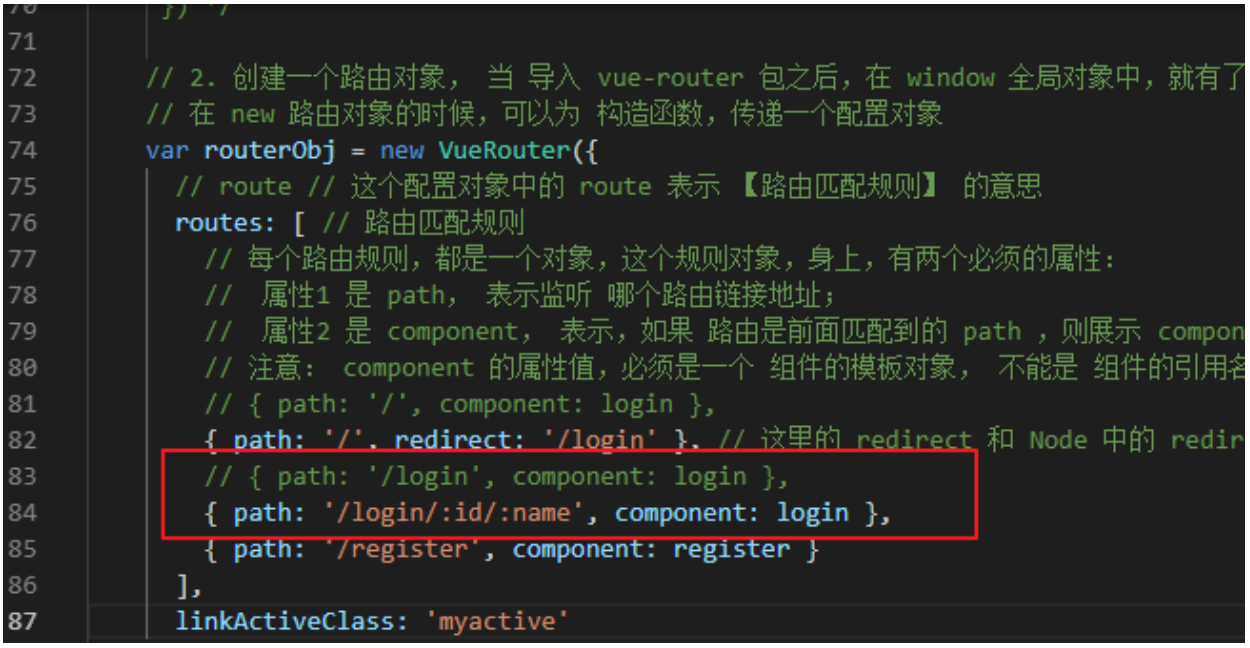
用 / 分割传参

<router-link>中的to 是跳转的地址

```
<!-- <a href="#/register">注册</a> -->

<!-- router-link 默认渲染为一个a 标签 -->
<!-- <router-link to="/login?id=10&name=ls" tag="span">登录</router-link> -->
<router-link to="/login/12/ls" tag="span">登录</router-link>
<router-link to="/register">注册</router-link>
```

```
55
56    <script>
57      // 组件的模板对象
58      var login = {
59      //   template: '<h1>登录{{$route.query.id}}==========={{$route.query.name}}组件</h1>'
60      template: '<h1>登录{{$route.params.id}}==========={{$route.params.name}}组件</h1>'
61    }
62
63      var register = {
64        template: '<h1>注册组件</h1>'
65      }
66
67
```

组件模板中 通过params去获取参数，而不是query



路由规则中路由的地址也有区别，如果传递参数个数不一致，则会访问不到

地址中要 :属性名/：属性名 的方式去写