1/9 Vue Router

前后端路由：（1/9）

后端路由：
后端路由 –> 路由 –>（地址 –> / add 通过 / 后面的东西，告诉后台 要进行什么操作）

后端路由的问题：

- 1、切换页面的时候是跳转全局刷新页面（用户体验差）
- 2、A页面的静态资源和B页面的静态资源会重复请求

SSR –> 服务器渲染，网络爬虫爬取资源的时候会及时找到重要的资源 –> 有利于是SEO优化，但是对服务器造成的压力较大，所以建议首页服务器渲染，别的还是走ajax。（在源代码中能在html部分显示）

前端路由：
通过不同的路由切换干不同的事或者切换不同的页面，
优势：用户所有的操作都在一个页面完成，单页应用（spa），既然是一个页面，共享资源只需要请求一次即可。

前端路由和后端路由：写法基本是一样的，有路由时，先通过服务器（判断里面有没有这个的操作）判断是不是后端路由，不是后端路由就是前端路由，写法上暂不明显区分。（自己总结）

vue中的路由：

新建项目的方法：(1/8、9)

Vue Router：创建过程（Vue Router是 –> vue中的路由管理器
）

第一种方法：自己配置：

- 1、安装：npm install vue-router

```
<!--下面都是在 router.js 中的内容 -->

// 第2步 引包
import Vue from 'vue'
```

```
import VueRouter form 'vue-router'
<!-- 在route.js中，先进行组件引入，定义引入的组件的名称，-->

// 第3步 使用（引用）
Vue.use(VuRouter)

// 第4步 需要配置router  ->  route.js
<!-- 在export default new Router中设置router，里面的内容为：path是路由内容，根据路由内容决定使用哪个组件：component：aa  （aa是组件名） -->

在new vue-router的时候，通过**routes** 来引配置数组，
new Vue的时候是通过router来引路由的。

// 5、放到根下面：
new Vue({
    el:'#app',
    render: () => h,
    router,
})
```

第二种方法：直接生成
用vue-cli 直接vue create 项目名称就搞定了。

- **vue create app （ app为项目名称 ）** –> 直接生成了名为APP的项目文件

配置路由表：(1/8)

配置路由表：配置什么路径就显示什么组件
‘/’ -> app
‘/ppa’ -> ppa

根据路由的实现方式不一样，配置的方式也不一样，详情见下面的路由的几种写法。

```
// 这是在router.js文件写的。

// 创建变量
const router = [
    {
        path:'/',
        component:app
    },
    {
        path:'./ppa',
        component:ppa
    }
]
```

```
// 导出这个变量，方便其他文件引入时使用
export default new VueRouter({
    routes
})
```

Vue Routes中的标签：(1/9)

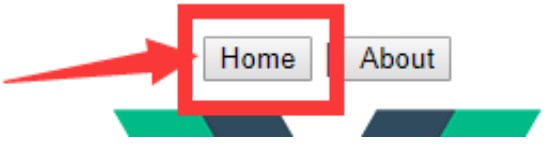
vue中的 `<router-link>` 标签：
`<router-link>`组件支持用户在具有路由功能的应用中 (点击) 导航。 通过 to 属性指定目标地址，默认渲染成带有正确链接的 `<a>` 标签，可以通过配置 tag 属性生成别的标签特性.。
这个标签和`<router-view>`标签是配套使用的，有`<router-view>`就有跳转后的内容，没有它就没有内容，它代表的是组件的内容。

`<toute-link>` 自动转成a标签
to='/' 放一个简单的字符串即可
除了可以成to，还可以是 " : to "
to的后面可是是一个字符串也可以是一个对象

如果是" : to " ，那么路径要写成字符串 ：to=" 'a' " 和to是有区别的。

```
// 例如：
<router-link to="/" tag="button">Home</router-link>
```

效果图：



`<router-view/>` 标签：代表组件内容的标签
可以认为是组件放的位置，如果不写就无法显示组件的内容。只要使用组件，就必须要写这个标签。

路由的写法：(1/9)

嵌套式路由：

常见的根据路由内容配置相应的组件：

```
let routes = [
  {
```

```
    path: '/',
    component: Home
  },
  {
    path: '/about',
    component:About,
    children:[
      {
        path:'a',
        name:'aaa',
        component:Aaa
      },
      {
        path:'b',
        name:'bbb',
        component:Bbb
      },
      {
        path:'c',
        component:Ccc
      }
    ]
  }
]

// 这是一个嵌套路由。先判断路由内容与那个path的内容相同，然后配置相应的组件，页面就会显示相应的内容。

// about里面还有一层是因为：这个是两层的选项卡，所以在第二个里面又嵌套了一层。
```

动态路由：

动态路由：path: '/:num'

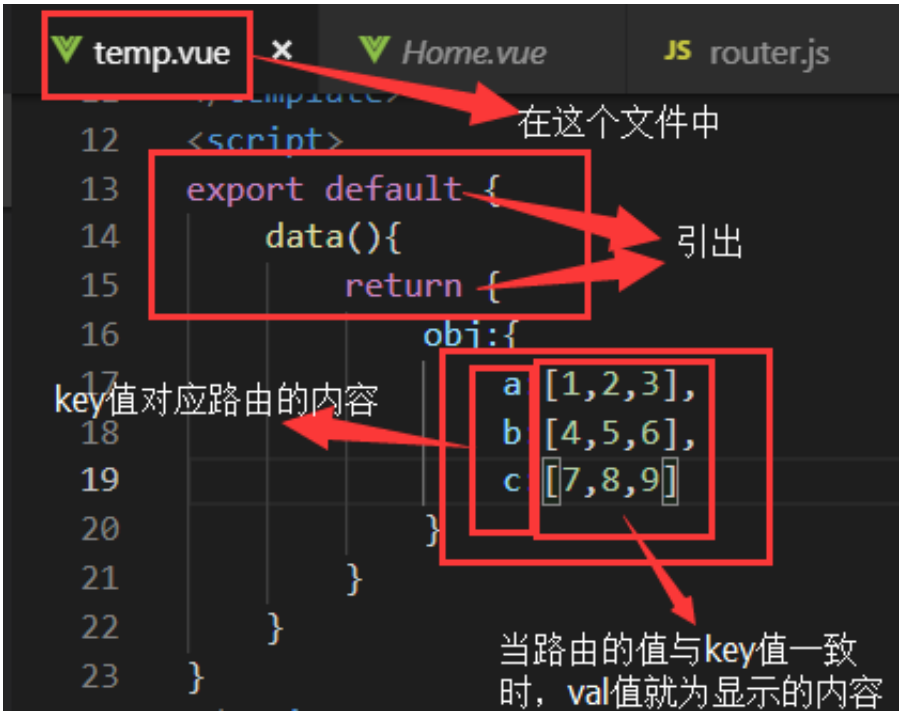
意思是：只要是根下的任意路径都能匹配到，通过**$route.params.num**来获取，

```
let routes = [
    {
        path:'/:num',
        component: temp    //  --> temp代表temp这个文件，根据判断路由里面的内容，来决定显示的内容。
    }
]

// 动态路由：寻找根目录的所有路径，

// 问题：就算能找到所有路径，难道路由的内容就是文件名？   那样的话temp又是什么？
```

```
// 答：在文件内会判断路由的内容，根据相应的内容显示相应的数据
```



编程式导航：

编程式路由可以使用 this.$router身上的方法。去任意切换路由，
主要方法（三种）：

- push（ ） –> 至直接放入跳转的路径(由)
  - 比如：this.$router.push(‘/c’)
  - 是追加，（往里面添加）
- go(num) –> 去第几个。负为后退，正为前进。
  - 比如：this.$router.go(-1) –> 返回上一步
  - this.$router.go(1) –> 返回下一步
- replace( ) 把当前路径替换成什么
  - 是替换（当前的被替换了，就没有了，只剩下替换后的了）

```
// 简单的应用一下
```

```
<template>
    <div>
        我是b1
        <button
            @click="changeRoute"
        >切到小宝宝这儿</button>
    </div>
</template>
<script>
export default {
    name:'b1',
    methods:{
        changeRoute(){
            this.$router.push('/c');
        }
    }
}
</script>
```

多层路由

Vue Routes 中有多层路由

如果有子路由，路由上挂一个children属性。为一个数组
数组下又包一些对象，对象的path不用加 “/”

有子级路由，记得在父级身上加上 `<router-view/>`
此时组件就会在父级的组件上显示。不管有多少层都遵循这个规律。

解决激活状态切换时，加不了事件。

```
//多层的代码示例：

let routes = [
  {
    path:'/a',
    component:Aaa
  },
  {
    path:'/b',
    component:Bbb,
    children:[
      {
        path:'/b/b1',
        component:b1
      }
    ]
```

```
  },
  {
    path:'/c',
    component:Ccc
  },
]
```

路由重定向：（1/11）

本来想要去A的然后去了B。然后路由显示的是去到的那个路由地址。而不是本来应该去的A地址。

重定向：redirect：后面是跳转的内容或文件

```
// 第一种写法：
{
    path:'/'
    redirect:'/xx'
}
// 当访问某个路径的时候，跳转到另一个路径中去。

// 第二种写法：
{
    path:'/',
    redirect:{name: xxx}
// 可以通过name 指定一个组件
// 需要在被转的组件里面写上：name:xxx

// 第三种写法：详细写法看文档。
{path: '/a',redirect:to =>{
    to：获取从哪里来的一个对象，记入了上一次路由的信息。
    常用：params、query、hash、path
    retrun  重定向的 字符串和路径。
}}
```