快速理解：React-router-dom

使用react+vite快速创建react项目

创建一个react-router的项目

```
npm create vite@latest react-router --template reace
```
js复制代码

```
yarn create vite react-router --template react
```
js复制代码

安装react-router-dom路由

- 因为使用的vite安装，所以是还未下载项目依赖需要先npm i 或者 yarn
- 再到在新建的项目文件夹终端运行 **npm install react-router-dom** 安装路由

路由方式BrowserRouter（history）和HashRouter（Hash）

import {BrowserRouter} from 'react-router-dom' 通过引入路由方式

- BrowserRouter:内核为浏览器的history,经过二次封账成BrowserRouter。
- HashRouter:根据url的#哈希值的变化。

「hash和history只有url是否拼接#号的区别。这里就用BrowserRouter模式。」

简单理解使用

在src文件夹下的App.jsx中引入路由

```
import {BrowserRouter,Routes,Route,Link} from 'react-router-dom'
```
js复制代码

BrowserRouter：包裹路由组件，声明使用history模式。

Link：是一个跳转的链接。经过编译后是 `<a>` 标签。

Routes：路由注册表用来包裹同一层级路由。

Route：注册路由，根据url的变化映射对应的element组件。

```
<BrowserRouter>
  <Link to='/'>主页</Link>
  <Link to='/about?id=123'>关于</Link>
  <Routes>
    <Route path='/' element={<Home></Home>}></Route>
    <Route path='/about' element={<About></About>}></Route>
  </Routes>
</BrowserRouter>
```

js复制代码

当使用Link进行跳转时直接进行参数拼接即可to='/about?id=123'

使用的hooks函数，编程试跳转传参

```
import {useNavigate} from 'react-router-dom'
```

js复制代码

useNavigate:只可以在已注册的路由组件中引入跳转到其他路由组件。

我们在Home的函数中调用： `const Navigate = useNavigate()`

在事件的回调函数中执行： `Navigate( /about?id=${msg} )` 就会引起路由的切换

接收参数

```
import {useSearchParams} from 'react-router-dom'
```

js复制代码

useSearchParams：接收url中以问号拼接的参数。

在引入这个hooks函数后调用 `const [params] = useSearchParams()`

使用 `params.get('id')` 把参数的键传入，就可以拿到参数的值

「以上是问号拼接的方式」

用‘/’路径形式传参的方法

例如： `to='/about/123'`

我们需要在注册路由时提前声明好，about后用斜杠拼接的是参数。

```
<Route path='/about/:id' element={<About></About>}></Route>
```


做好声明后就可以直接使用斜杠拼接参数了

```
<button onClick={()=>navgete(`/about/${msg}`)}>发送到关于</button>
```


使用字符串的模板拼接语法，在/后拼接了一个msg

如何获取拼接的参数？和问号拼接的不同。

```
import {useParams} from 'react-router-dom'
```


useParams:获取url'/'路径拼接参数的hooks函数

`const params = useParams()`

`params.id` 获取msg的值

出现的问题

我们发现，这个路由组件和静态页面写到一起的话非常繁杂，不易于维护。如果有新入职的程序员看见堆积如山的代码，是个什么样的心情呢？ 所以我们使用模块化开发。我们把路由组件提取出来形成一个新组件再和主页面放在一起。就会变得有层次感，写出优雅的代码。

我们创建一个路由层。routes文件夹index.jsx。

```
import {useRoutes} from 'react-router-dom'
```


useRoutes：路由注册表hooks函数。

```
import Home from '../pages/home'
import About from '../pages/about'
const routes = [
  {
    path:'/',
    element:<Home/>
  },
  {
    path:'/',
    element:<About/>
  },
  {
    path:'*',
    element:<NotFround/>
  }
]

function element(){
  const el = useRoutes(routes)
  return el
}

export default element
```

useRoutes()根据url的变化返回一个路由组件。

我们再到App.jsx中引入抛出的函数

路由方式的包裹BrowserRouter还是不能少的。

「以上是学习react-router-dom的个人理解。感谢」