React Router V6.4 基础知识（地基篇）

如果你想了解 react-router-dom 的背后原理，你可能需要知道背后的基础知识，对基础知识的重视才是最重要的，阅读这篇文章，帮助阅读者节省时间整理资料。

history 对象

- 属性

属性名	说明
length	回一个整数（Integer），该整数表示会话历史中元素的数目，包括当前加载的页。
scrollRestoration	允许 Web 应用程序在历史导航上显式地设置默认滚动恢复行为。
state	返回一个表示历史堆栈顶部的状态的任意（any）值

- 方法

方法名	说明
back()	等价于 `history.go(-1)`。
forward()	history.go(1)
go()	通过当前页面的相对位置从浏览器历史记录（会话记录）异步加载页面。
pushState()	按指定的名称和 URL（如果提供该参数）将数据 push 进会话历史栈，数据被 DOM 进行不透明处理；
replaceState()	按指定的数据、名称和 URL（如果提供该参数），更新 history 栈上最新的条目。

- 说明

history 对象的核心功能就是跳转，修改 url, 保存 state 和滚动位置。

location 对象

- 属性

属性名	说明
Location.href	完整的 url
Location.protocol	协议
Location.host	域名可能带有端口号
Location.hostname	URL 域名
Location.port	端口号
Location.pathname	以 "/" 开头的路径
Location.search	以"?"开头的URL参数
Location.hash	以“#”开头的 hash
Location.username	包含 URL 中域名前的用户名
Location.password	包含 URL 域名前的密码
Location.origin	包含页面来源的域名的标准形式

- 方法

方法名	说明
Location.assign()	加载给定 URL 的内容资源到这个 Location 对象所关联的对象上。
Location.reload()	重新加载来自当前 URL 的资源。
Location.replace()	用给定的 URL 替换掉当前的资源。
Location.toString()	返回一个DOMString，包含整个 URL。

- 说明

location 对象的诸多属性，重要是 pathname/search/hash 三个属性非常重要。

小结

在 react-router v6.4 中 history 和 location 被融合到 history 文件中，得到一个全新的 history 对象，这个全新的 history 对象是 react-router 的基础。在 React-Router v5 时代 history 还被单独抽象成了一个 npm 包, 提供了适合同意的 api 接口。

URL 构造器

创建并返回一个 `URL` 对象

- 属性

属性名	方法
hash	包含 # 的字符串
host	主机名
hostname	URL 域
href	完整的 URL
origin	包含协议名、地址、端口
password	域名密码
pathname	以 / 开头的路径
port	URL端口
protocol	包含 url 协议名
search	以 ? 开始字符串
searchParams	URLSearchParams对象
username	包含在域名前面指定的用户名

- 方法

属性名	方法
toString()	返回包含整个 URL， 不能用于修改值
toJSON()	返回包含整个 URL
reateObjectURL()	包含一个唯一的 blob 链接
revokeObjectURL()	销毁之前使用URL.createObjectURL()方法创建的 URL 实例

URLSearchParams

URLSearchParams 接口定义了一些实用的方法来处理 URL 的查询字符串。

- 方法

方法名	说明
URLSearchParams.append()	插入一个指定的键/值对作为新的搜索参数。
URLSearchParams.delete()	从搜索参数列表里删除指定的搜索参数及其对应的值。
URLSearchParams.entries()	返回一个iterator可以遍历所有键/值对的对象。
URLSearchParams.get()	获取指定搜索参数的第一个值。
URLSearchParams.getAll()	获取指定搜索参数的所有值，返回是一个数组。
URLSearchParams.has()	返回 Boolean 判断是否存在此搜索参数。
URLSearchParams.keys()	返回iterator 此对象包含了键/值对的所有键名。
URLSearchParams.set()	设置一个搜索参数的新值，假如原来有多个值将删除其他所有的值。
URLSearchParams.sort()	按键名排序。
URLSearchParams.toString()	返回搜索参数组成的字符串，可直接使用在 URL 上。
URLSearchParams.values()	返回iterator 此对象包含了键/值对的所有值。

如果我们的有一个 params 字符串，要转换成 javascript 对象访问属性，此构造函数非常有用。

web 事件

- popstate 事件

> 调用 history.pushState() 或者 history.replaceState() 不会触发 popstate 事件

```js
window.onpopstate = function(event) {
  alert("location: " + document.location + ", state: " + JSON.stringify(event.state));
};

// 使用 pushState api
let state = {page: 1}
```

```
history.pushState(state, "title 1", "?page=1");
state.page = 2;
history.pushState(state, "title 2", "?page=2");
// 使用 back()/forward()/go() 三个 api 触发 popstate 事件
history.back()
history.go(-1)
```

- hashchange 事件

hashchange 就比较简单，showcode

```
window.addEventListener('hashchange', function() {
  console.log('The hash has changed!')
}, false);
```

fetch api

实现了以前的 xhr 的 api，

- fetch

```
fetch('http://your_api.com/content.json')
  .then(response => response.json())
  .then(data => console.log(data));
```

- Request

属性名	说明
Request.method	包含请求的方法 (GET, POST, 等.)
Request.url	包含这个请求的 URL。
Request.headers	包含请求相关的Headers对象。
Request.context	包含请求的上下文 (例如：audio, image, iframe, 等)
Request.referrer	?包含请求的来源 (例如：client)。
Request.referrerPolicy	?包含请求来源的策略 (例如：no-referrer)。
Request.mode	包含请求的模式 (例如： cors, no-cors, same-origin, navigate).Request.credentials 只读
Request.credentials	包含请求的证书 (例如： omit, same-origin).
Request.redirect	包含？如何处理重定向模式，它可能是一个follow，error或者manual。
Request.integrity	包含请求的子资源的完整性值 (例如： sha256-BpfBw7ivV8q2jLiT13fxDYAe2tJllusRSZ273h2nFSE=).
Request.cache	包含请求的缓存模式 (例如： default, reload, no-cache).

- Response

Response.headers	包含此 Response 所关联的 Headers 对象。
Response.ok	包含了一个布尔值，标示该 Response 成功（HTTP 状态码的范围在 200-299）。
Response.redirected	表示该 Response 是否来自一个重定向，如果是的话，它的 URL 列表将会有多个条目。
Response.status	包含 Response 的状态码（例如 200 表示成功）。
Response.statusText	包含了与该 Response 状态码一致的状态信息（例如，OK 对应 200）。
Response.type	包含 Response 的类型（例如，basic、cors）。
Response.url	包含 Response 的 URL。

虽然在实际前端开发中，我们很少能用到 Request 和 Response 两种构造函数，但是在 React Router 中确实用到了。

FormData

一个简单的例子

```js
let formData = new FormData();

// 添加属性
formData.append("name", "magnesuim-")

// 使用 xhr 对象将formdat 发给送出去
var request = new XMLHttpRequest();
request.open("POST", "http://your_api/content");
request.send(formData);
```

- 方法

方法	属性名
FormData.append()	在 formData 添加字段
FormData.delete()	从 FormData 对象里面删除一个键值对。
FormData.entries()	返回一个包含所有键值对的iterator对象。
FormData.get()	返回在 FormData 对象中与给定键关联的第一个值。
FormData.getAll()	返回一个包含 FormData 对象中与给定键关联的所有值的数组。
FormData.has()	返回一个布尔值表明 FormData 对象是否包含某些键。
FormData.keys()	返回一个包含所有键的iterator对象。
FormData.set()	给 FormData 设置属性值，如果FormData 对应的属性值存在则覆盖原值，否则新增一项属性值。
FormData.values()	返回一个包含所有值的iterator对象。

小结

- 在开始 React Router DOM 源码之前，基础知识点需要巩固实践，业务逻辑要熟练熟悉。history 对象中添加了 state 相关的属性和方法，此方法和属性对实现 history 有很好的帮助。react-router-dom 中实现了一些之前版本没有的功能：如路由位置保存，路由之前获取数据的 loader 函数，以及获取表单的 action 函数等等，就 react-router 官方将 react-router v6.4 是值得。
- 坚实基础