## **1. React路由原理**

* 不同的路径渲染不同的组件
* 有两种实现方式
  + HashRouter:利用hash实现路由切换
  + BrowserRouter:实现h5 Api实现路由的切换

### **1.1 HashRouter**

* 利用hash实现路由切换

public\index.html

<!DOCTYPE html><html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<title>React App</title></head>

<body>

<div id="root"></div>

<a href="#/a">去a</a>

<a href="#/b">去b</a>

<script>

window.addEventListener('hashchange', () => {

console.log(window.location.hash);

});

</script></body></html>

### **1.2 BrowserRouter**

* 利用h5 Api实现路由的切换

#### **1.2.1 history**

* [history](https://developer.mozilla.org/zh-CN/docs/Web/API/Window/history)对象提供了操作浏览器会话历史的接口。
* historylength 属性声明了浏览器历史列表中的元素数量
* [pushState](https://developer.mozilla.org/zh-CN/docs/Web/API/History_API) HTML5引入了 history.pushState() 和 history.replaceState() 方法，它们分别可以添加和修改历史记录条目。这些方法通常与window.onpopstate配合使用
* [onpopstate](https://developer.mozilla.org/zh-CN/docs/Web/API/Window/onpopstate) window.onpopstate是popstate事件在window对象上的事件处理程序

#### **1.2.2 pushState**

* pushState会往History中写入一个对象，他造成的结果便是,History length +1、url 改变、该索引History对应有一个State对象,这个时候若是点击浏览器的后退，便会触发popstate事件，将刚刚的存入数据对象读出
* pushState 会改变History
* 每次使用时候会为该索引的State加入我们自定义数据
* 每次我们会根据State的信息还原当前的view，于是用户点击后退便有了与浏览器后退前进一致的感受
* pushState() 需要三个参数: 一个状态对象, 一个标题 (目前被忽略), 和 (可选的) 一个URL
* 调用history.pushState()或者history.replaceState()不会触发popstate事件. popstate事件只会在浏览器某些行为下触发, 比如点击后退、前进按钮(或者在JavaScript中调用history.back()、history.forward()、history.go()方法)

<!DOCTYPE html><html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<title>React App</title></head>

<body>

<div id="root"></div>

<script>

window.onpopstate = (event) => {

console.log({ state: event.state, pathname: window.location.pathname, type: 'popstate' });

};

window.onpushstate = (event) => {

console.log(event);

};

(**function** (history) {

**var** pushState = history.pushState;

history.pushState = **function** (state, title, pathname) {

**if** (**typeof** window.onpushstate == "function") {

window.onpushstate({ state, pathname, type: 'pushstate' });

}

**return** pushState.apply(history, arguments);

};

})(window.history);

//绑定事件处理函数.

setTimeout(() => {

history.pushState({ page: 1 }, "title 1", "/page1");

}, 1000);

setTimeout(() => {

history.pushState({ page: 2 }, "title 2", "/page2");

}, 2000);

setTimeout(() => {

history.replaceState({ page: 3 }, "title 3", "/page3");

}, 3000);

setTimeout(() => {

history.back();

}, 4000);

setTimeout(() => {

history.go(1);//向前进一步

}, 5000);

// page1 => page2 => page3 => page2 => page3

</script></body></html>

## **2. 跑通路由**

* [createBrowserHistory](https://github.com/ReactTraining/history/blob/master/modules/createBrowserHistory.js)
* [createHashHistory.js](https://github.com/ReactTraining/history/blob/master/modules/createHashHistory.js)

cnpm i react-router-dom @types/react-router-dom path-to-regexp -S

### **2.1 src\index.tsx**

src\index.tsx

**import** React **from** 'react';**import** ReactDOM **from** 'react-dom';**import** { HashRouter **as** Router, Route } **from** 'react-router-dom';**import** Home **from** './components/Home';**import** User **from** './components/User';**import** Profile **from** './components/Profile';

ReactDOM.render(

<Router>

<div>

<Route path="/" component={Home} />

<Route path="/user" component={User} />

<Route path="/profile" component={Profile} />

</div>

</Router>

, document.getElementById('root'));

### **2.2 Home.tsx**

src\components\Home.tsx

**import** React, { Component } **from** 'react';**export** **default** **class** **Home** **extends** **Component** {

render() {

**return** (

<div>Home</div>

)

}

}

### **2.3 User.tsx**

src\components\User.tsx

**import** React, { Component } **from** 'react';**import** { RouteComponentProps } **from** '../react-router-dom';

interface Params { }

type Props = RouteComponentProps<Params> & {

}**export** **default** **class** **User** **extends** **Component** {

render() {

console.log(**this**.props);

**return** (

<div>User</div>

)

}

}

/\*\*

{

history: H.History;

location: H.Location<S>;

match: match<Params>;

staticContext?: C;

}

export interface Location<S = LocationState> {

pathname: Pathname;

state: S;

}

export interface match<Params extends { [K in keyof Params]?: string } = {}> {

params: Params;

isExact: boolean;

path: string;

url: string;

}

\*/



**function** **createBrowserHistory**(props = {}) {

**const** globalHistory = window.history;

**let** listeners = [];

**function** **createHref**(location) {

**let** { path, search, hash } = location;

**return** path + search + hash;

}

**function** **setState**(nextState) {

Object.assign(history, nextState);

history.length = globalHistory.length;

listeners.forEach(listener => listener());

}

**function** **push**(path, state) {

**const** action = 'PUSH';

**const** location = { state, path };

globalHistory.pushState(state, null, path);

setState({ action, location });

}

**function** **replace**(path, state) {

**const** action = 'REPLACE';

**const** location = { state, path };

globalHistory.replaceState(state, null, path);

setState({ action, location });

}

**function** **go**(n) {

globalHistory.go(n);

}

**function** **goBack**() {

go(-1);

}

**function** **goForward**() {

go(1);

}

**let** isBlocked = false;

**function** **block**(prompt = false) {

isBlocked = prompt;

}

**function** **listen**(listener) {

listeners.push(listener);

}

**const** history = {

length: globalHistory.length,

action: 'POP',

location: { pathname: window.location.pathname, state: globalHistory.state },

createHref,

push,

replace,

go,

goBack,

goForward,

block,

listen

};

**return** history;

}

**let** myHistory = createBrowserHistory();console.log('myHistory', myHistory);

### **2.4 Profile.tsx**

src\components\Profile.tsx

**import** React, { Component } **from** 'react';**export** **default** **class** **Profile** **extends** **Component** {

render() {

**return** (

<div>Profile</div>

)

}

}

## **3. 实现路由**

### **3.1 history\index.tsx**

src\history\index.tsx

**export** \* **from** './types';

### **3.2 history\types.tsx**

src\history\types.tsx

**export** interface Location {

pathname: string;

state?: any;

}**export** interface History {

location: Location;

}

### **3.3 react-router-dom\index.tsx**

src\react-router-dom\index.tsx

**import** HashRouter **from** './HashRouter';**import** Route **from** './Route';**export** {

HashRouter,

Route

}**export** \* **from** './types';

### **3.4 react-router-dom\types.tsx**

src\react-router-dom\types.tsx

**import** { History } **from** '../history';**export** type Location = History['location'];**export** interface ContextValue {

location?: Location

}**export** interface match<Params = {}> {

params: Params;

isExact: boolean;

path: string;

url: string;

}**export** interface RouteComponentProps<Params = {}> {

history: History;

location: Location;

match: match<Params>;

}

### **3.5 react-router-dom\context.tsx**

src\react-router-dom\context.tsx

**import** { ContextValue } **from** './types';**import** { createContext } **from** 'react';**export** **default** createContext<ContextValue>({});

### **3.6 HashRouter.tsx**

src\react-router-dom\HashRouter.tsx

**import** React, { Component } **from** 'react'**import** Context **from** './context';**import** { ContextValue, Location } **from** './types';

interface Props { }

interface State {

location: Location;

}**export** **default** **class** **HashRouter** **extends** **Component**<**Props**, **State**> {

state = {

location: {

pathname: window.location.hash.slice(1),

state: null

}

}

componentWillMount() {

window.addEventListener('hashchange', (event: HashChangeEvent) => {

**this**.setState({

location: {

...this.state.location,

pathname: window.location.hash.slice(1) || '/'

}

});

});

window.location.hash = window.location.hash || '/';

}

render(): React.ReactNode {

**let** value: ContextValue = {

location: **this**.state.location

}

**return** (

<Context.Provider value={value}>

{this.props.children}

</Context.Provider>

)

}

}

### **3.7 Route.tsx**

src\react-router-dom\Route.tsx

**import** React, { Component,ComponentType } **from** 'react';**import** RouterContext **from** './context';

interface Props {

path: string;

component: ComponentType<any>

}**export** **default** **class** **Route** **extends** **Component**<**Props**> {

**static** contextType = RouterContext;

render() {

**let** { path, component: Component } = **this**.props;

**let** pathname = **this**.context.location.pathname;

**if** (pathname.startsWith(path)) {

**return** <Component />

} **else** {

**return** null;

}

}

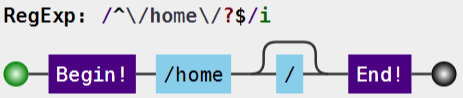
}

## **4.path-to-regexp**

* [path-to-regexp](https://www.npmjs.com/package/path-to-regexp)
* [regulex](https://jex.im/regulex)

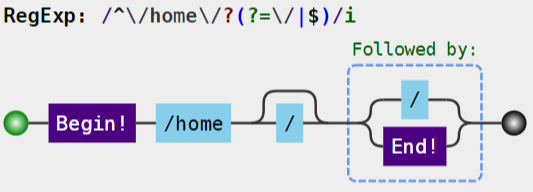
### **4.1 /home结束**

**let** pathToRegExp = require('path-to-regexp');**let** regxp = pathToRegExp('/home',[],{end:true});console.log(regxp);// /^\/home\/?$/iconsole.log(regxp.test('/home'));console.log(regxp.test('/home/2'));



### **4.2 /home非结束**

**let** pathToRegExp = require('path-to-regexp');**let** regx2 = pathToRegExp('/home',[],{end:false});console.log(regx2);// /^\/home\/?(?=\/|$)/iconsole.log(regx2.test('/home'));console.log(regx2.test('/home/'));console.log(regx2.test('/home//'));console.log(regx2.test('/home/2'));



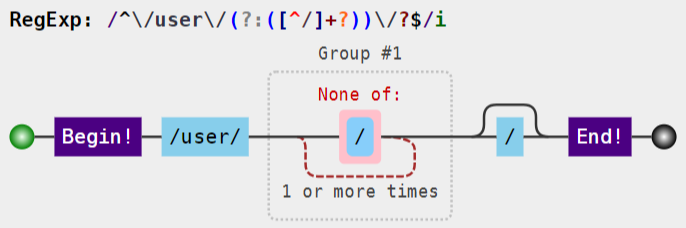
### **4.3 路径参数**

**let** params = [];**let** regx3 = pathToRegExp('/user/:id',params,{end:true});console.log(regx3,params);/\*\*

/^\/user\/(?:([^\/]+?))\/?$/i

[ { name: 'id', optional: false, offset: 7 } ]

\*\*/



### **4.4 正则匹配**

| **表达式** | **含义** |
| --- | --- |
| () | 表示捕获分组，()会把每个分组里的匹配的值保存起来，使用$n(n是一个数字，表示第n个捕获组的内容) |
| (?:) | 表示非捕获分组，和捕获分组唯一的区别在于，非捕获分组匹配的值不会保存起来 |

* 正则匹配的前瞻就是给正则匹配的选项定义一个断言，或者说是一个条件比如：我要匹配一个字母，但是我的需求是字母后面必须是跟着一个数字的情况，那么这种场景是怎么实现了，就是用到前瞻的概念，那么我想要他的前面也要是一个数字怎么办了，这就是后顾

| **表达式** | **含义** |
| --- | --- |
| (?=pattern) | 正向肯定查找(前瞻),后面必须跟着什么 |
| (?!pattern) | 正向否定查找(前瞻)，后面不能跟着什么 |
| (?<=pattern) | 反向肯定条件查找(后顾),不捕获 |
| (?<!pattern) | 反向否定条件查找（后顾） |

console.log('1a'.match(/\d(?=[a-z])/));console.log('1@'.match(/\d(?![a-z])/));console.log('a1'.match(/(?<=[a-z])\d/));console.log('$1'.match(/(?<![a-z])\d/));

## **5.正则匹配**

### **5.1 Route.js**

src\react-router-dom\Route.js

import React, { Component } from 'react';

import RouterContext from './context';+import { pathToRegexp } from 'path-to-regexp';

interface Props {

path: string;+ exact?: boolean;

component: ComponentType<any>

}

export default class Route extends Component<Props> {

static contextType = RouterContext;

render() {+ let { path="/", component: RouteComponent, exact = false } = this.props;+ let pathname = this.context.location.pathname;+ let regxp = pathToRegexp(path, [], { end: exact });+ let result = pathname.match(regxp);+ if (result) {

return <RouteComponent />

} else {

return null;

}

}

}

## **6. 实现Link**

### **6.1 src\index.tsx**

src\index.tsx

import React from 'react';

import ReactDOM from 'react-dom';+import { HashRouter as Router, Route, Link } from './react-router-dom';

import Home from './components/Home';

import User from './components/User';

import Profile from './components/Profile';

ReactDOM.render(

<Router>

<div>+ <ul>+ <li><Link to="/">Home</Link></li>+ <li><Link to="/user">User</Link></li>+ <li><Link to="/profile">Profile</Link></li>+ </ul>+ <Route path="/" component={Home} exact />

<Route path="/user" component={User} />

<Route path="/profile" component={Profile} />

</div>

</Router>

, document.getElementById('root'));

### **6.2 history\types.tsx**

src\history\types.tsx

export interface Location {

pathname: string;

state?: any;

}

export interface History {

location: Location;+ push(path: string, state?: any): void;

}

### **6.3 react-router-dom\types.tsx**

src\react-router-dom\types.tsx

import { History } from '../history';

export type Location = History['location'];

export interface ContextValue {

location?: Location;+ history?: History

}

export interface match<Params = {}> {

params: Params;

isExact: boolean;

path: string;

url: string;

}

export interface RouteComponentProps<Params = {}> {

history: History;

location: Location;

match?: match<Params>;

}

### **6.4 Link.tsx**

src\react-router-dom\Link.tsx

**import** React, { Component } **from** 'react'**import** RouterContext **from** './context';**import** { LocationDescriptor } **from** '../history';**export** interface LinkProps {

to: LocationDescriptor;

}**export** **default** **class** **Link** **extends** **Component**<**LinkProps**> {

**static** contextType = RouterContext;

render() {

**return** (

<a {...this.props} onClick={() => this.context.history.push(this.props.to)}>{this.props.children}</a>

)

}

}

### **6.5 HashRouter.tsx**

src\react-router-dom\HashRouter.tsx

import React, { Component } from 'react'

import Context from './context';

import { ContextValue } from './types';+import { LocationDescriptor, Location } from '../history';

interface Props { }

interface State {

location: Location;

}

export default class HashRouter extends Component<Props, State> {

locationState: any

state = {

location: {+ pathname: window.location.hash.slice(1),+ state: null

}

}

componentWillMount() {

window.addEventListener('hashchange', (event: HashChangeEvent) => {

this.setState({

location: {

...this.state.location,

pathname: window.location.hash.slice(1) || '/',+ state: this.locationState

}

});

});

window.location.hash = window.location.hash || '/';

}

render(): React.ReactNode {+ let that = this;

let value: ContextValue = {

location: this.state.location,

history: {

location: this.state.location,+ push(to: LocationDescriptor) {+ if (typeof to === 'object') {+ let { pathname, state } = to;+ that.locationState = state;+ window.location.hash = pathname!;+ } else {+ window.location.hash = to;+ }

}

}

}

return (

<Context.Provider value={value}>

{this.props.children}

</Context.Provider>

)

}

}

## **7. 引入bootstrap**

cnpm i bootstrap@3 -S

### **7.1 src\index.tsx**

src\index.tsx

import React from 'react';

import ReactDOM from 'react-dom';

import { HashRouter as Router, Route, Link } from './react-router-dom';

import Home from './components/Home';

import User from './components/User';

import Profile from './components/Profile';+import 'bootstrap/dist/css/bootstrap.css';

ReactDOM.render(

<Router>+ <>+ <div className="navbar navbar-inverse">+ <div className="container-fluid">+ <div className="navbar-heading">+ <div className="navbar-brand">珠峰架构</div>+ </div>+ <ul className="nav navbar-nav">+ <li><Link to="/">Home</Link></li>+ <li><Link to="/user">User</Link></li>+ <li><Link to="/profile">Profile</Link></li>+ </ul>+ </div>+ </div>+ <div className="container">+ <div className="row">+ <div className="col-md-12">+ <Route path="/" exact component={Home} />+ <Route path="/user" component={User} />+ <Route path="/profile" component={Profile} />+ </div>+ </div>+ </div>+ </>

</Router>

, document.getElementById('root'));

## **8. Redirect&Switch**

### **8.1 src\index.tsx**

src\index.tsx

import React from 'react';

import ReactDOM from 'react-dom';+import { HashRouter as Router, Route, Link, Redirect, Switch } from './react-router-dom';

import Home from './components/Home';

import User from './components/User';

import Profile from './components/Profile';

import 'bootstrap/dist/css/bootstrap.css';

ReactDOM.render(

<Router>

<>

<div className="navbar navbar-inverse">

<div className="container-fluid">

<div className="navbar-heading">

<div className="navbar-brand">珠峰架构</div>

</div>

<ul className="nav navbar-nav">

<li><Link to="/">Home</Link></li>

<li><Link to="/user">User</Link></li>

<li><Link to="/profile">Profile</Link></li>

</ul>

</div>

</div>

<div className="container">

<div className="row">

<div className="col-md-12">+ <Switch>+ <Route path="/" exact component={Home} />+ <Route path="/user" component={User} />+ <Route path="/profile" component={Profile} />+ <Redirect to="/" />+ </Switch>

</div>

</div>

</div>

</>

</Router>

, document.getElementById('root'));

### **8.2 react-router-dom\index.tsx**

src\react-router-dom\index.tsx

import HashRouter from './HashRouter';

import Route from './Route';

import Link from './Link';+import Switch from './Switch';+import Redirect from './Redirect';

export {

HashRouter,

Route,

Link,+ Switch,+ Redirect

}

export \* from './types';

### **8.3 Switch.tsx**

src\react-router-dom\Switch.tsx

**import** React, { Component } **from** 'react'**import** Context **from** './context';**import** { pathToRegexp } **from** 'path-to-regexp';

interface Props {

children: Array<JSX.Element>

}**export** **default** **class** **Switch** **extends** **Component**<**Props**> {

**static** contextType = Context;

render() {

**let** pathname = **this**.context.location.pathname;

**if** (**this**.props.children) {

**for** (**let** i = 0; i < **this**.props.children.length; i++) {

**let** child: JSX.Element = **this**.props.children[i];

**let** { path = '/', component: Component, exact = false } = child.props;

**let** regxp = pathToRegexp(path, [], { end: exact });

**let** result = pathname.match(regxp);

**if** (result) {

**return** child;

}

}

}

**return** null;

}

}

### **8.4 Redirect.tsx**

src\react-router-dom\Redirect.tsx

**import** React, { Component } **from** 'react'**import** { LocationDescriptor } **from** '../history';

interface Props {

to: LocationDescriptor;

}**export** **default** **class** **extends** **Component**<**Props**> {

**static** contextType = Context;

render() {

**this**.context.history.push(**this**.props.to);

**return** null;

}

}

## **9. 路径参数**

### **9.1 User.tsx**

src\components\User.tsx

import React, { Component } from 'react';+import { RouteComponentProps, Link, Route } from '../react-router-dom';+import UserAdd from './UserAdd';+import UserDetail from './UserDetail';+import UserList from './UserList';

interface Params { }

type Props = RouteComponentProps<Params> & {

}

export default class User extends Component {

render() {

return (+ <div className="row">+ <div className="col-md-2">+ <ul className="nav nav-stack">+ <li><Link to="/user/list">用户列表</Link></li>+ <li><Link to="/user/add">添加用户</Link></li>+ </ul>+ </div>+ <div className="col-md-10">+ <Route path="/user/add" component={UserAdd} />+ <Route path="/user/list" component={UserList} />+ <Route path="/user/detail/:id" component={UserDetail} />+ </div>+ </div>+ )

}

}

/\*\*

{

history: H.History;

location: H.Location<S>;

match: match<Params>;

staticContext?: C;

}

export interface Location<S = LocationState> {

pathname: Pathname;

search: Search;

state: S;

hash: Hash;

key?: LocationKey;

}

export interface match<Params extends { [K in keyof Params]?: string } = {}> {

params: Params;

isExact: boolean;

path: string;

url: string;

}

\*/

### **9.2 UserAdd.tsx**

src\components\UserAdd.tsx

**import** React, { Component, RefObject } **from** 'react';**import** { RouteComponentProps } **from** '../react-router-dom';

type Props = RouteComponentProps;**export** **default** **class** **UserAdd** **extends** **Component**<**Props**> {

usernameRef: RefObject<HTMLInputElement>

**constructor**(props: Props) {

**super**(props);

**this**.usernameRef = React.createRef<HTMLInputElement>();

}

handleSubmit = (event: React.FormEvent<HTMLFormElement>) => {

event.preventDefault();

**let** username = **this**.usernameRef.current!.value;

**let** usersStr = localStorage.getItem('users');

**let** users = usersStr ? JSON.parse(usersStr) : [];

users.push({ id: Date.now() + '', username });

localStorage.setItem('users', JSON.stringify(users));

**this**.props.history.push('/user/list');

}

render() {

**return** (

<form onSubmit={this.handleSubmit}>

<input className="form-control" type="text" ref={this.usernameRef} />

<button type="submit" className="btn btn-primary">提交</button>

</form>

)

}

}

### **9.3 src\types.tsx**

src\types.tsx

**export** interface User {

id?: string;

username?: string;

}

### **9.4 UserList.tsx**

src\components\UserList.tsx

**import** React, { Component } **from** 'react'**import** { Link, RouteComponentProps } **from** '../react-router-dom';**import** { User } **from** '../types';

type Props = RouteComponentProps;

interface State {

users: Array<User>

}

**export** **default** **class** **UserList** **extends** **Component**<**Props**, **State**> {

state = { users: [] }

componentDidMount() {

**let** usersStr = localStorage.getItem('users');

**let** users: Array<User> = usersStr ? JSON.parse(usersStr) : [];

**this**.setState({ users });

}

render() {

**return** (

<ul className="list-group">

{

this.state.users.map((user: User, index) => (

<li className="list-group-item" key={index}>

<Link to={{ pathname: `/user/detail/${user.id}`, state: user }}>{user.username}</Link>

</li>

))

}

</ul>

)

}

}

### **9.5 UserDetail.tsx**

src\components\UserDetail.tsx

**import** React, { Component } **from** 'react';**import** { RouteComponentProps } **from** '../react-router-dom';**import** { User } **from** '../types';

interface Params {

id: string;

}

type Props = RouteComponentProps<Params, any, User> & {

}

interface State {

user: User

}**export** **default** **class** **UserDetail** **extends** **Component**<**Props**, **State**> {

state = {

user: {}

}

componentDidMount() {

**let** user: User = **this**.props.location.state!;

**if** (!user) {

**let** usersStr = localStorage.getItem('users');

**let** users = usersStr ? JSON.parse(usersStr) : [];

**let** id = **this**.props.match!.params.id;

user = users.find((user: User) => user.id === id);

}

**if** (user) **this**.setState({ user });

}

render() {

**let** user: User = **this**.state.user;

**return** (

<div>

{user.id}:{user.username}

</div>

)

}

}

### **9.6 history\types.tsx**

src\history\types.tsx

+export interface Location<T = any> {+ pathname: string;+ state?: T;+}+export type LocationDescriptor = string | Location;+export interface History<T = any> {+ location: Location<T>;+ push(pathname: string): void;+ push(to: Location): void;+}

### **9.7 Link.tsx**

src\react-router-dom\Link.tsx

import React, { Component } from 'react'

import RouterContext from './context';+import { LocationDescriptor } from '../history';

interface Props {+ to: LocationDescriptor;

}

export default class Link extends Component<Props> {

static contextType = RouterContext;

render() {

return (

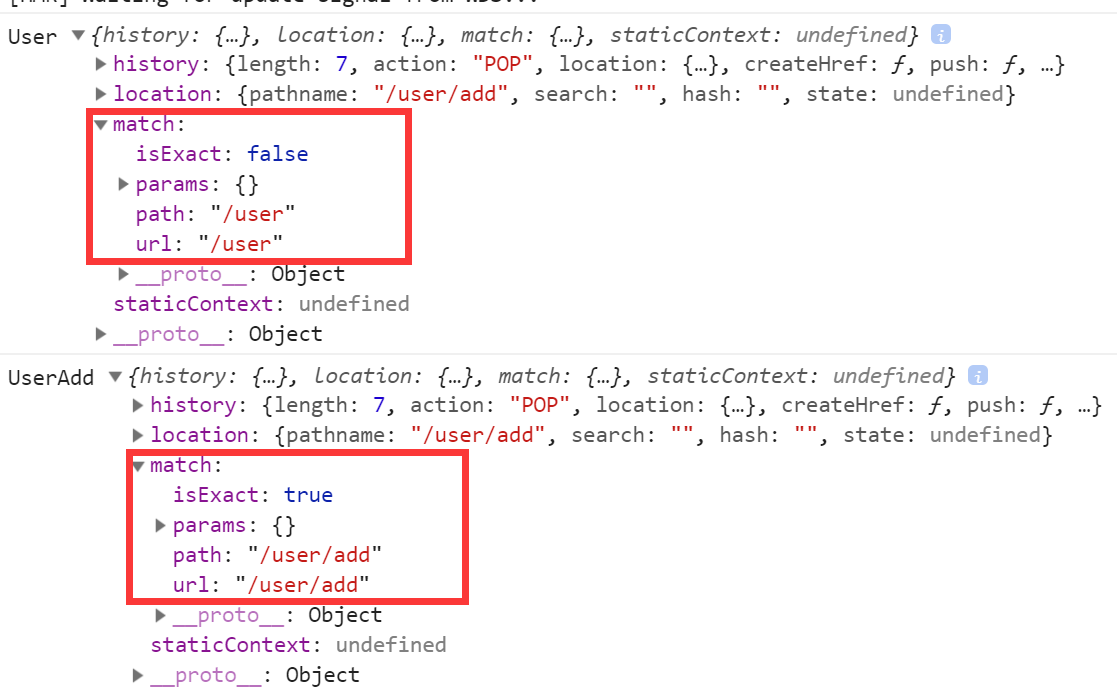
<a onClick={() => this.context.history.push(this.props.to)}>{this.props.children}</a>

)

}

}

### **9.8 Route.tsx**



src\react-router-dom\Route.tsx

import React, { Component,ComponentType } from 'react';

import RouterContext from './context';+import { RouteComponentProps, match } from './';+import { pathToRegexp, Key } from 'path-to-regexp';

interface Props {

path: string;

exact?: boolean;+ component: ComponentType<RouteComponentProps<any>>

}

export default class Route extends Component<Props> {

static contextType = RouterContext;

render() {

let { path, component: RouteComponent, exact = false } = this.props;

let pathname = this.context.location.pathname;+ let keys: Array<Key> = [];+ let regxp = pathToRegexp(path, keys, { end: exact });

let result = pathname.match(regxp);

if (result) {+ let [url, ...values] = result;+ let paramNames = keys.map((item: Key) => item.name);+ let memo: Record<string, any> = {};+ //values=['zhufeng',10] paramNames=['name','age']+ let params = values.reduce((memo: Record<string, any>, val: string, index: number) => {+ memo[paramNames[index]] = val;+ return memo;+ }, memo);+ type ParamsType = typeof params;+ //当路由路径和当前路径成功匹配，会生成一个match对象+ let matchResult: match<ParamsType> = {+ url, //URL中匹配到的部分+ path, //用来的路径正则，取自path属性+ isExact: pathname === url,//判断匹配到的url是否和完整路径匹配+ params+ }+ let props: RouteComponentProps<ParamsType> = {+ location: this.context.location,+ history: this.context.history,+ match: matchResult+ }+ return <RouteComponent {...props} />;+ }

return null;

}

}

### **9.9 react-router-dom\types.tsx**

src\react-router-dom\types.tsx

import { Location, History } from '../history';

export interface ContextValue {

location?: Location;

history?: History

}

export interface match<Params = {}> {

params: Params;

isExact: boolean;

path: string;

url: string;

}+export interface RouteComponentProps<Params = {}, staticContext = {}, State = any> {+ history: History;+ location: Location<State>;+ match?: match<Params>;+}

### **9.10 Home.tsx**

src\components\Home.tsx

import React, { Component } from 'react';+import { RouteComponentProps } from '../react-router-dom';+type Props = RouteComponentProps;+export default class Home extends Component<Props> {

render() {

return (

<div>Home</div>

)

}

}

### **9.11 Profile.tsx**

src\components\Profile.tsx

import React, { Component } from 'react';+import { RouteComponentProps } from '../react-router-dom';+type Props = RouteComponentProps;+export default class Profile extends Component<Props> {

render() {

return (

<div>Profile</div>

)

}

}

## **10. 受保护的路由**

### **10.1 src\index.tsx**

src\index.tsx

import React from 'react';

import ReactDOM from 'react-dom';

import { HashRouter as Router, Route, Link, Redirect, Switch } from './react-router-dom';

import Home from './components/Home';

import User from './components/User';

import Profile from './components/Profile';+import Protected from './components/Protected';+import Login from './components/Login';

import 'bootstrap/dist/css/bootstrap.css';

ReactDOM.render(

<Router>

<>

<div className="navbar navbar-inverse">

<div className="container-fluid">

<div className="navbar-heading">

<div className="navbar-brand">珠峰架构</div>

</div>

<ul className="nav navbar-nav">

<li><Link to="/">Home</Link></li>

<li><Link to="/user">User</Link></li>

<li><Link to="/profile">Profile</Link></li>

</ul>

</div>

</div>

<div className="container">

<div className="row">

<div className="col-md-12">

<Switch>

<Route path="/" exact component={Home} />

<Route path="/user" component={User} />+ <Route path="/login" component={Login} />+ <Protected path="/profile" component={Profile} />

<Redirect to="/" />

</Switch>

</div>

</div>

</div>

</>

</Router>

, document.getElementById('root'));

### **10.2 Protected.tsx**

src\components\Protected.tsx

**import** React **from** 'react'**import** { Route, Redirect } **from** '../react-router-dom';

interface Props extends Record<string, any> {

path: string;

component: React.ComponentType<any>;

}**export** **default** (props: Props) => {

**let** { component: RouteComponent, path } = props;

**return** (

<Route path={path} render={

(props: any) => (

localStorage.getItem('logined') ? <RouteComponent {...props} /> : <Redirect to={{ pathname: '/login', state: { from: props.location.pathname } }} />

)

} />

)

}

### **10.3 Login.tsx**

src\components\Login.tsx

**import** React, { Component } **from** 'react';**import** { RouteComponentProps } **from** '../react-router-dom';

type Props = RouteComponentProps;**export** **default** **class** **Login** **extends** **Component**<**Props**> {

handleClick = () => {

localStorage.setItem('logined', 'true');

**if** (**this**.props.location.state)

**this**.props.history.push(**this**.props.location.state.from);

}

render() {

**return** (

<button className="btn btn-primary" onClick={this.handleClick}>登录</button>

)

}

}

### **10.4 Route.tsx**

src\react-router-dom\Route.tsx

import React, { Component, ComponentType } from 'react';

import RouterContext from './context';

import { RouteComponentProps, match } from './';

import { pathToRegexp, Key } from 'path-to-regexp';

interface Props {

path?: string;

exact?: boolean;

component?: ComponentType<RouteComponentProps<any>>;+ render?: (props: any) => React.ReactNode

}

export default class Route extends Component<Props> {

static contextType = RouterContext;

render() {+ let { path="/", component: RouteComponent, exact = false, render } = this.props;

let pathname = this.context.location.pathname;

let keys: Array<Key> = [];

let regxp = pathToRegexp(path, keys, { end: exact });

let result = pathname.match(regxp);

if (result) {

let [url, ...values] = result;

let paramNames = keys.map((item: Key) => item.name);

let memo: Record<string, any> = {};

let params = values.reduce((memo: Record<string, any>, val: string, index: number) => {

memo[paramNames[index]] = val;

return memo;

}, memo);

type ParamsType = typeof params;

let matchResult: match<ParamsType> = {

url: pathname,

isExact: pathname === url,

path,

params

}

let props: RouteComponentProps<ParamsType> = {

location: this.context.location,

history: this.context.history,

match: matchResult

}+ if (RouteComponent)+ return <RouteComponent {...props} />;+ else if (render) {+ return render(props);+ } else {+ return null;+ }

}

return null;

}

}

## **11. 自定义导航**

### **11.1 src\index.tsx**

src\index.tsx

import React from 'react';

import ReactDOM from 'react-dom';+import { HashRouter as Router, Route, MenuLink, Redirect, Switch } from './react-router-dom';

import Home from './components/Home';

import User from './components/User';

import Profile from './components/Profile';

import Protected from './components/Protected';

import Login from './components/Login';

import 'bootstrap/dist/css/bootstrap.css';

ReactDOM.render(

<Router>

<>

<div className="navbar navbar-inverse">

<div className="container-fluid">

<div className="navbar-heading">

<div className="navbar-brand">珠峰架构</div>

</div>

<ul className="nav navbar-nav">+ <li><MenuLink exact to="/">Home</MenuLink></li>+ <li><MenuLink exact to="/user">User</MenuLink></li>+ <li><MenuLink exact to="/profile">Profile</MenuLink></li>

</ul>

</div>

</div>

<div className="container">

<div className="row">

<div className="col-md-12">

<Switch>

<Route path="/" exact component={Home} />

<Route path="/user" component={User} />

<Route path="/login" component={Login} />

<Protected path="/profile" component={Profile} />

<Redirect to="/" />

</Switch>

</div>

</div>

</div>

</>

</Router>

, document.getElementById('root'));

### **11.2 react-router-dom\index.tsx**

src\react-router-dom\index.tsx

import HashRouter from './HashRouter';

import Route from './Route';

import Link from './Link';

import Switch from './Switch';

import Redirect from './Redirect';+import MenuLink from './MenuLink';

export {

HashRouter,

Route,

Link,

Switch,

Redirect,+ MenuLink

}

export \* from './types';

### **11.3 MenuLink.tsx**

src\react-router-dom\MenuLink.tsx

import React from 'react'

import { Route, Link } from '.';

import './MenuLink.css';

import { LocationDescriptor } from '../history';

import { match } from '.';

interface Props {

to: LocationDescriptor,

exact?: boolean;

children?: React.ReactNode

}

export default (props: Props) => {

let { to, exact, children } = props;

return <Route

path={typeof to === 'object' ? to.pathname! : to}

exact={exact}

children={

(childProps: any) => (

<Link className={childProps.match ? 'active' : ''} to={to} {...childProps}>{children}</Link>

)

}

/>

}

### **11.4 MenuLink.css**

src\react-router-dom\MenuLink.css

.navbar-inverse .navbar-nav > li > .active{

background-color: green!important;

color:red!important;

}

## **12. withRouter**

### **12.1 src\index.tsx**

import React from 'react';

import ReactDOM from 'react-dom';

import { HashRouter as Router, Route, MenuLink, Redirect, Switch } from './react-router-dom';

import Home from './components/Home';

import User from './components/User';

import Profile from './components/Profile';

import Protected from './components/Protected';

import Login from './components/Login';+import NavHeader from './components/NavHeader';

import 'bootstrap/dist/css/bootstrap.css';

ReactDOM.render(

<Router>

<>

<div className="navbar navbar-inverse">

<div className="container-fluid">+ <NavHeader title="欢迎来到珠峰架构" />

<ul className="nav navbar-nav">

<li><MenuLink exact to="/">Home</MenuLink></li>

<li><MenuLink exact to="/user">User</MenuLink></li>

<li><MenuLink exact to="/profile">Profile</MenuLink></li>

</ul>

</div>

</div>

<div className="container">

<div className="row">

<div className="col-md-12">

<Switch>

<Route path="/" exact component={Home} />

<Route path="/user" component={User} />

<Route path="/login" component={Login} />

<Protected path="/profile" component={Profile} />

<Redirect to="/" />

</Switch>

</div>

</div>

</div>

</>

</Router>

, document.getElementById('root'));

### **12.2 NavHeader.tsx**

src\components\NavHeader.tsx

**import** React **from** 'react';**import** { RouteComponentProps } **from** '../react-router-dom';**import** { withRouter } **from** '../react-router-dom';//只有当一个组件是通过路由Route渲染出来的话才会有RouteComponentProps里的属性

interface NavHeaderProps {

title: string;

}

**class** **NavHeader** **extends** **React**.**Component**<**RouteComponentProps** & **NavHeaderProps**> {

render() {

**return** (

<div className="navbar-header">

<div

onClick={(event: React.MouseEvent) => this.props.history.push('/')}

className="navbar-brand">{this.props.title}</div>

</div>

)

}

}**export** **default** withRouter<NavHeaderProps>(NavHeader);

### **12.3 withRouter.tsx**

src\react-router-dom\withRouter.tsx

**import** React **from** 'react';**import** { Route, RouteComponentProps } **from** './';**export** **default** **function** <**NavHeaderProps**>(OldComponent: React.ComponentType<NavHeaderProps & RouteComponentProps>) {

**return** (props: NavHeaderProps) => (

<Route render={

(routeProps: RouteComponentProps) => <OldComponent {...props} {...routeProps} />

} />

)

}

### **12.4 react-router-dom\index.tsx**

src\react-router-dom\index.tsx

import HashRouter from './HashRouter';

import Route from './Route';

import Link from './Link';

import Switch from './Switch';

import Redirect from './Redirect';

import MenuLink from './MenuLink';+import withRouter from './withRouter';

export {

HashRouter,

Route,

Link,

Switch,

Redirect,

MenuLink,+ withRouter

}

export \* from './types';

## **13. 阻止跳转**

### **13.1 UserAdd.tsx**

src\components\UserAdd.tsx

import React, { Component, RefObject } from 'react';+import { RouteComponentProps, Prompt } from '../react-router-dom';

type Props = RouteComponentProps;+interface State {+ isBlocking: boolean;+}+export default class UserAdd extends Component<Props, State> {+ state = {+ isBlocking: false+ }

usernameRef: RefObject<HTMLInputElement>

constructor(props: Props) {

super(props);

this.usernameRef = React.createRef<HTMLInputElement>();

}

handleSubmit = (event: React.FormEvent<HTMLFormElement>) => {

event.preventDefault();+ this.setState({+ isBlocking: false+ }, () => {+ let username = this.usernameRef.current!.value;+ let usersStr = localStorage.getItem('users');+ let users = usersStr ? JSON.parse(usersStr) : [];+ users.push({ id: Date.now() + '', username });+ localStorage.setItem('users', JSON.stringify(users));+ this.props.history.push('/user/list');+ })

}

render() {+ let { isBlocking } = this.state;

return (

<form onSubmit={this.handleSubmit}>+ <Prompt+ when={isBlocking}+ message={location => `你确定要跳转到${location.pathname}吗？`}+ />+ <input className="form-control" type="text" ref={this.usernameRef} onChange={(event: React.ChangeEvent<HTMLInputElement>) => {+ this.setState({ isBlocking: event.target.value.length > 0 });+ }} />

<button type="submit" className="btn btn-primary">提交</button>

</form>

)

}

}

### **13.2 history\types.tsx**

src\history\types.tsx

+import { Message } from '../react-router-dom';

export interface Location<T = any> {

pathname: string;

state?: T;

}

export type LocationDescriptor = string | Location;

export interface History<T = any> {

push(pathname: string): void;

push(to: Location): void;+ message:Message | null+ block(message: Message | null): void;

}

### **13.3 react-router-dom\types.tsx**

src\react-router-dom\types.tsx

import { Location, History } from '../history';

export interface ContextValue {

location?: Location;

history?: History

}

export interface match<Params = {}> {

params: Params;

isExact: boolean;

path: string;

url: string;

}

export interface RouteComponentProps<Params = {}, staticContext = {}, State = any> {

history: History;

location: Location<State>;

match?: match<Params>;

}+export interface Message {+ (location: Location): string+}

### **13.4 react-router-dom\Prompt.tsx**

src\react-router-dom\Prompt.tsx

**import** React **from** 'react'**import** RouterContext **from** './context';**import** { History } **from** '../history';**import** { Message } **from** './';

interface Props {

when: boolean;

message: Message;

}**export** **default** **class** **Prompt** **extends** **React**.**Component**<**Props**> {

**static** contextType = RouterContext;

history: History

componentWillUnmount() {

**this**.history.block(null);

}

render() {

**this**.history = **this**.context.history;

**const** { when, message } = **this**.props;

**if** (when) {

**this**.history.block(message);

} **else** {

**this**.history.block(null);

}

**return** null;

}

}

### **13.5 react-router-dom\HashRouter.tsx**

src\react-router-dom\HashRouter.tsx

import React, { Component } from 'react'

import Context from './context';

import { ContextValue } from './types';

import { LocationDescriptor, Location } from '../history';

import { Message } from './';

interface Props { }

interface State {

location: Location;

}

export default class HashRouter extends Component<Props, State> {

locationState: any

prompt: Message | null

state = {

location: {

pathname: window.location.hash.slice(1),

state: null

}

}

componentWillMount() {

window.addEventListener('hashchange', (event: HashChangeEvent) => {

this.setState({

location: {

...this.state.location,

pathname: window.location.hash.slice(1) || '/',

state: this.locationState

}

});

});

window.location.hash = window.location.hash || '/';

}

render(): React.ReactNode {

let that = this;

let value: ContextValue = {

location: this.state.location,

history: {

push(to: LocationDescriptor) {+ if (that.prompt) {+ let allow = window.confirm(that.prompt(typeof to === 'object' ? to as Location : { pathname: to }));+ if (!allow) return;+ }

if (typeof to === 'object') {

let { pathname, state } = to;

that.locationState = state;

window.location.hash = pathname!;

} else {

window.location.hash = to;

}

},+ prompt : null,+ block(prompt : Message | null) {+ that.prompt = prompt ;+ }

}

}

return (

<Context.Provider value={value}>

{this.props.children}

</Context.Provider>

)

}

}

### **13.6 react-router-dom\index.tsx**

src\react-router-dom\index.tsx

import HashRouter from './HashRouter';

import Route from './Route';

import Link from './Link';

import Switch from './Switch';

import Redirect from './Redirect';

import MenuLink from './MenuLink';

import withRouter from './withRouter';+import Prompt from './Prompt';

export {

HashRouter,

Route,

Link,

Switch,

Redirect,

MenuLink,

withRouter,+ Prompt

}

export \* from './types';

## **14. BrowserRouter**

### **14.1 public\index.html**

public\index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" href="%PUBLIC\_URL%/favicon.ico" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<meta name="theme-color" content="#000000" />

<title>React App</title>+ <script>+ (function (history) {+ var pushState = history.pushState;+ history.pushState = function (state, title, pathname) {+ if (typeof window.onpushstate == "function") {+ window.onpushstate(state, pathname);+ }+ return pushState.apply(history, arguments);+ };+ })(window.history);+ </script>

</head>

<body>

<div id="root"></div>

</body>

</html>

### **14.2 src\index.tsx**

src\index.tsx

import ReactDOM from 'react-dom';+import { BrowserRouter as Router, Route, MenuLink, Redirect, Switch } from './react-router-dom';

import Home from './components/Home';

### **14.3 BrowserRouter.tsx**

src\react-router-dom\BrowserRouter.tsx

**import** React, { Component } **from** 'react'**import** Context **from** './context';**import** { Message } **from** './';**import** { LocationDescriptor, Location } **from** '../history';

declare global {

interface Window {

onpushstate: (state: any, pathname: string) => **void**;

}

}

**export** **default** **class** **BrowserRouter** **extends** **Component** {

state = {

location: { pathname: '/' }

}

message: Message | null

componentDidMount() {

window.onpopstate = (event: PopStateEvent) => {

**this**.setState({

location: {

...this.state.location,

pathname: document.location.pathname,

state: event.state

}

});

};

window.onpushstate = (state: any, pathname: string) => {

**this**.setState({

location: {

...this.state.location,

pathname,

state

}

});

};

}

render() {

**let** that = **this**;

**let** value = {

location: that.state.location,

history: {

push(to: LocationDescriptor) {

**if** (that.message) {

**let** allow = window.confirm(that.message(**typeof** to == 'object' ? to : { pathname: to }));

**if** (!allow) **return**;

}

**if** (**typeof** to === 'object') {

**let** { pathname, state } = to;

window.history.pushState(state, '', pathname);

} **else** {

window.history.pushState('', '', to);

}

},

block(message: Message) {

that.message = message;

}

}

}

**return** (

<Context.Provider value={value}>

{this.props.children}

</Context.Provider>

)

}

}

### **14.4 react-router-dom\index.tsx**

src\react-router-dom\index.tsx

import HashRouter from './HashRouter';

import Route from './Route';

import Link from './Link';

import Switch from './Switch';

import Redirect from './Redirect';

import MenuLink from './MenuLink';

import withRouter from './withRouter';

import Prompt from './Prompt';+import BrowserRouter from './BrowserRouter';

export {

HashRouter,

Route,

Link,

Switch,

Redirect,

MenuLink,

withRouter,

Prompt,+ BrowserRouter

}

export \* from './types';

## **参考**

### **可选参数**

**let** express = require("express");**let** app = express();/\* app.get('/member/?:path/?:tag', (req, res) => {

res.json(req.params);

}); \*/// /^\/member\/?(?:([^\/]+?))\/?(?:([^\/]+?))\/?$/i**let** reg = /^\/member\/([^\/]+?)?(?:\/([^\/]+?))?\/?$/;

app.get(reg, (req, res) => {

res.json({ path: req.params[0], tag: req.params[1] });

});

app.listen(9999);// /member/path