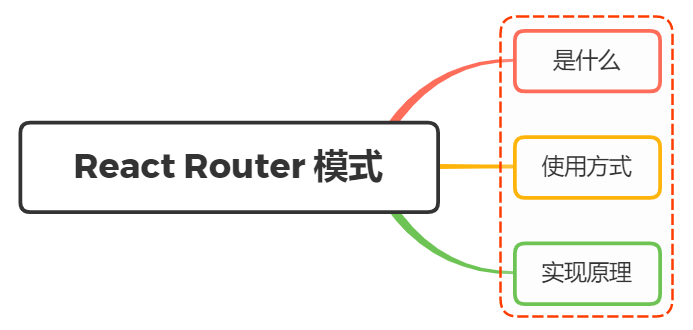
# 面试官：说说React Router有几种模式？实现原理？



## 一、是什么

在单页应用中，一个web项目只有一个html页面，一旦页面加载完成之后，就不用因为用户的操作而进行页面的重新加载或者跳转，其特性如下：

* 改变 url 且不让浏览器像服务器发送请求
* 在不刷新页面的前提下动态改变浏览器地址栏中的URL地址

其中主要分成了两种模式：

* hash 模式：在url后面加上#，如http://127.0.0.1:5500/home/#/page1
* history 模式：允许操作浏览器的曾经在标签页或者框架里访问的会话历史记录

## 二、使用

React Router对应的hash模式和history模式对应的组件为：

* HashRouter
* BrowserRouter

这两个组件的使用都十分的简单，作为最顶层组件包裹其他组件，如下所示

// 1.import { BrowserRouter as Router } from "react-router-dom";  
// 2.import { HashRouter as Router } from "react-router-dom";  
  
import React from 'react';  
import {  
 BrowserRouter as Router,  
 // HashRouter as Router   
 Switch,  
 Route,  
} from "react-router-dom";  
import Home from './pages/Home';  
import Login from './pages/Login';  
import Backend from './pages/Backend';  
import Admin from './pages/Admin';  
  
  
function App() {  
 return (  
 <Router>  
 <Route path="/login" component={Login}/>  
 <Route path="/backend" component={Backend}/>  
 <Route path="/admin" component={Admin}/>  
 <Route path="/" component={Home}/>  
 </Router>  
 );  
}  
  
export default App;

## 三、实现原理

路由描述了 URL 与 UI之间的映射关系，这种映射是单向的，即 URL 变化引起 UI 更新（无需刷新页面）

下面以hash模式为例子，改变hash值并不会导致浏览器向服务器发送请求，浏览器不发出请求，也就不会刷新页面

hash 值改变，触发全局 window 对象上的 hashchange 事件。所以 hash 模式路由就是利用 hashchange 事件监听 URL 的变化，从而进行 DOM 操作来模拟页面跳转

react-router也是基于这个特性实现路由的跳转

下面以HashRouter组件分析进行展开：

## HashRouter

HashRouter包裹了整应用，

通过window.addEventListener('hashChange',callback)监听hash值的变化，并传递给其嵌套的组件

然后通过context将location数据往后代组件传递，如下：

import React, { Component } from 'react';  
import { Provider } from './context'  
// 该组件下Api提供给子组件使用  
class HashRouter extends Component {  
 constructor() {  
 super()  
 this.state = {  
 location: {  
 pathname: window.location.hash.slice(1) || '/'  
 }  
 }  
 }  
 // url路径变化 改变location  
 componentDidMount() {  
 window.location.hash = window.location.hash || '/'  
 window.addEventListener('hashchange', () => {  
 this.setState({  
 location: {  
 ...this.state.location,  
 pathname: window.location.hash.slice(1) || '/'  
 }  
 }, () => console.log(this.state.location))  
 })  
 }  
 render() {  
 let value = {  
 location: this.state.location  
 }  
 return (  
 <Provider value={value}>  
 {  
 this.props.children  
 }  
 </Provider>  
 );  
 }  
}  
  
export default HashRouter;

### Router

Router组件主要做的是通过BrowserRouter传过来的当前值，通过props传进来的path与context传进来的pathname进行匹配，然后决定是否执行渲染组件

import React, { Component } from 'react';  
import { Consumer } from './context'  
const { pathToRegexp } = require("path-to-regexp");  
class Route extends Component {  
 render() {  
 return (  
 <Consumer>  
 {  
 state => {  
 console.log(state)  
 let {path, component: Component} = this.props  
 let pathname = state.location.pathname  
 let reg = pathToRegexp(path, [], {end: false})  
 // 判断当前path是否包含pathname  
 if(pathname.match(reg)) {  
 return <Component></Component>  
 }  
 return null  
 }  
 }  
 </Consumer>  
 );  
 }  
}  
export default Route;

## 参考文献

* https://juejin.cn/post/6870376090297171975#heading-9
* https://segmentfault.com/a/1190000023560665