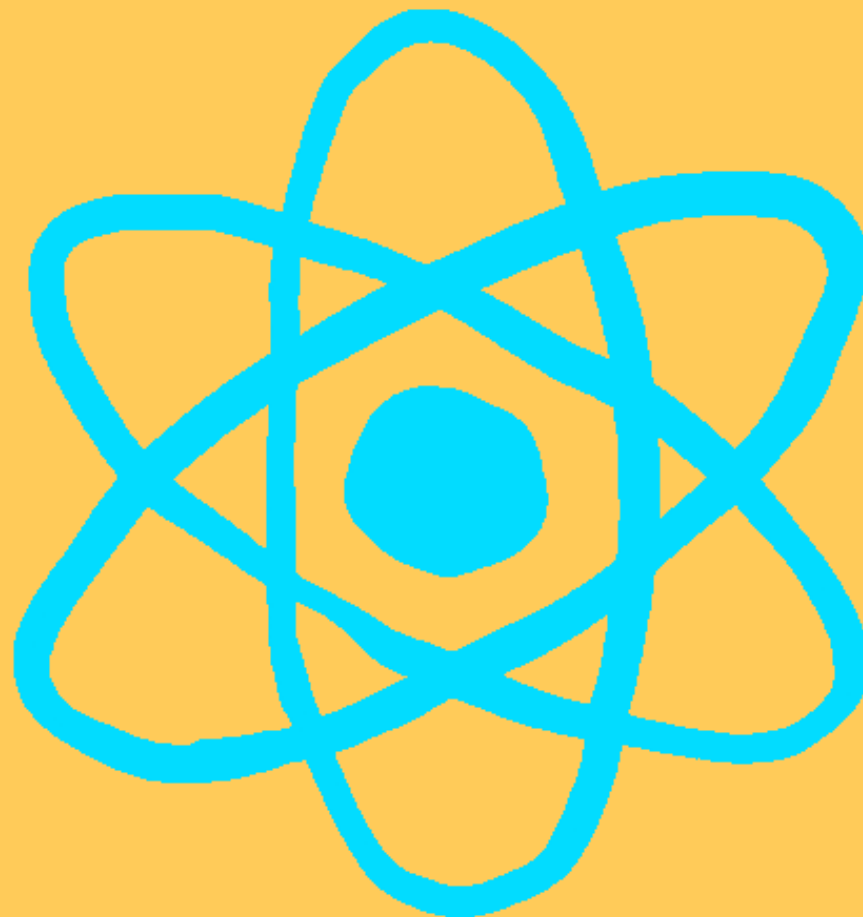


FRONTEND. REACT. ЛЕКЦІЯ 8




REACT ROUTER V6. REACT.LAZY. SUSPENSE



REACT ROUTER

<https://reactrouter.com/en/main>

The screenshot shows the top of the React Router website. At the top left is the React Router logo and the text "React Router". To its right is the word "main" with a dropdown arrow, followed by a sun icon. Further right are icons for GitHub and Discord, and the text "Made by Remix 7". Below the header is a navigation bar with a chevron icon and the word "Navigation". The main content area is divided into four cards. The top-left card is titled "What's New in 6.4?" and features a bar chart icon. The top-right card is titled "I'm New" and features the React Router logo icon. The bottom-left card is titled "I'm on v5" and features a migration icon. The bottom-right card is titled "I'm Stuck!" and features a question mark icon.

React Router main    Made by Remix 7

> **Navigation**

What's New in 6.4?
v6.4 is our most exciting release yet with new data abstractions for reads, writes, and navigation hooks to easily keep your UI in sync with your data. The new feature overview will catch you up.

I'm New
Start with the tutorial. It will quickly introduce you to the primary features of React Router: from configuring routes, to loading and mutating data, to pending and optimistic UI.

I'm on v5
The migration guide will help you migrate

I'm Stuck!
Running into a problem? Chances are

REACT ROUTER

```
npm install react-router-dom
```

REACT ROUTER

REACT ROUTER

- react-router-dom

REACT ROUTER

- react-router-dom
- react-router-native

REACT ROUTER

- `react-router-dom`
- `react-router-native`
- `react-router`

REACT ROUTER

REACT ROUTER

- v6

REACT ROUTER

- v6
- spa navigation

ТИПИ РОУТЕРІВ

ТИПИ РОУТЕРІВ

- BrowserRouter

ТИПИ РОУТЕРІВ

- BrowserRouter
- MemoryRouter

ТИПИ РОУТЕРІВ

- BrowserRouter
- MemoryRouter
- HashRouter

ТИПИ РОУТЕРІВ

- BrowserRouter
- MemoryRouter
- HashRouter
- NativeRouter

ТИПИ РОУТЕРІВ

- BrowserRouter
- MemoryRouter
- HashRouter
- NativeRouter
- StaticRouter

BROWSERROUTER

```
import './App.css';

import { Lesson1 } from './pages/Lesson1';
import { Home } from './pages/Home';
import { PageNotFound } from './pages/404';
import { Articles } from './pages/Articles';

import { Routes, Route, BrowserRouter } from 'react-router-dom';
import { Layout } from './components/Layout';
import { Header } from './components/Header';

function App() {

return (
  <div>
    <header favcolor="blue"></header>
    <browserrouter>
```

V6.4 ТИПИ РОУТЕРІВ

V6.4 ТИПИ РОУТЕРІВ

- createBrowserRouter

V6.4 ТИПИ РОУТЕРІВ

- `createBrowserRouter`
- `createMemoryRouter`

V6.4 ТИПИ РОУТЕРІВ

- createBrowserRouter
- createMemoryRouter
- createHashRouter

CREATEBROWSERROUTER

```
import './App.css';

import { Lesson1 } from './pages/Lesson1';
import { Home } from './pages/Home';
import { PageNotFound } from './pages/404';
import { Articles } from './pages/Articles';

import { Routes, createRoutesFromElements, Route, createBrowserRouter, RouterProvider } from 'react-router-dom';
import { Layout } from './components/Layout';
import { Header } from './components/Header';

function App() {
  const router = createBrowserRouter(
    createRoutesFromElements(
      <route path="/" element="{<Layout>}">
        <route path="/" element="{<Home>}">Home</route>
        <route path="/lesson1" element="{<Lesson1>}">Home</route>
        <route path="/articles/:id" element="{<Articles>}">Home</route>
      </route>
    )
  );
}
```

CREATEBROWSERROUTER

CREATEBROWSERROUTER

- Рекомендований маршрутизатор

CREATEBROWSERROUTER

- Рекомендований маршрутизатор
- Базується на DOM History API

ROUTES

Масив роутерів. Можливо використовувати, як структуру даних

```
createBrowserRouter([
  {
    path: "/",
    element: <root>,
    loader: rootLoader,
    children: [
      {
        path: "events/:id",
        element: <event>,
        loader: eventLoader,
      },
    ],
  },
]);
</event></root>
```

BASENAME

Властивість. Задання посилання на корінь ресурсу

```
createBrowserRouter(routes, {  
  basename: "/app",  
});
```

ROUTE

```
const router = createBrowserRouter([
  {
    // it renders this element
    element: <team>,

    // when the URL matches this segment
    path: "teams/:teamId",

    // with this data loaded before rendering
    loader: async ({ request, params }) => {
      return fetch(
        `/fake/api/teams/${params.teamId}.json`,
        { signal: request.signal }
      );
    },

    // performing this mutation when data is submitted to it
    action: async ({ request }) => {
```

PATH

Властивість. Шлях для маршрутизації

```
path: "teams/:teamId",
```

PATH

Динамічні сегменти. Не можуть бути частковими.

```
path: "teams/:teamId/product/:productId",
```

CATCHALL *

Все після сегменту

```
path: "teams/:teamId/product/*",
```

LINK

Навігація на клієнті

```
<header classname="header-menu">  
  <link to="/">Home  
  <link to="/lesson1">Lesson1  
  <link to="/sdfdfdsf">Incorrect  
  <link to="/articles">Articles  
</header>
```


LINK

LINK

- to

LINK

- to
- relative

LINK

- to
- relative
- preventScrollReset

NAVLINK

```
<header classname="header-menu">  
  <navlink to="/">Home</navlink>  
  <navlink to="/lesson1">Lesson1</navlink>  
  <navlink to="/sdfdfdsf">Incorrect</navlink>  
</header>
```

OUTLET

Місце, куди буде доданий весь контент

INDEX

Мало б працювати, але...

```
const router = createBrowserRouter(  
  createRoutesFromElements(  
    <route path="/" element="{<Layout">}">  
      <route index="" element="{<Home">}">Home</route>  
      <route path="/lesson1" element="{<Lesson1">}">Home</route>  
      <route path="/articles/:id" element="{<Articles">}">Home</route>  
      <route path="*" element="{<PageNotFound">}">Home</route>  
    </route>  
  )  
);
```

USEPARAMS

```
import { useParams } from "react-router-dom";

const Articles = () => {
  const {id} = useParams();

  return (<div>
    <h1>Articles</h1>
    <span>Page: {id}</span>
  </div>)
}
export { Articles };
```


USENAVIGATE

Можливість емуляції браузерної навігації

```
import { useEffect, useState } from "react";
import { useParams, useNavigate } from "react-router-dom";

const Articles = () => {
  const {id} = useParams();
  const navigate = useNavigate();

  const [data, setData] = useState({});

  const goBack = () => navigate(-1);

  useEffect(() => {
    fetch(`https://jsonplaceholder.typicode.com/posts/${id}`)
      .then(response => response.json())
      .then(json => setData(json));
  }, []);
```

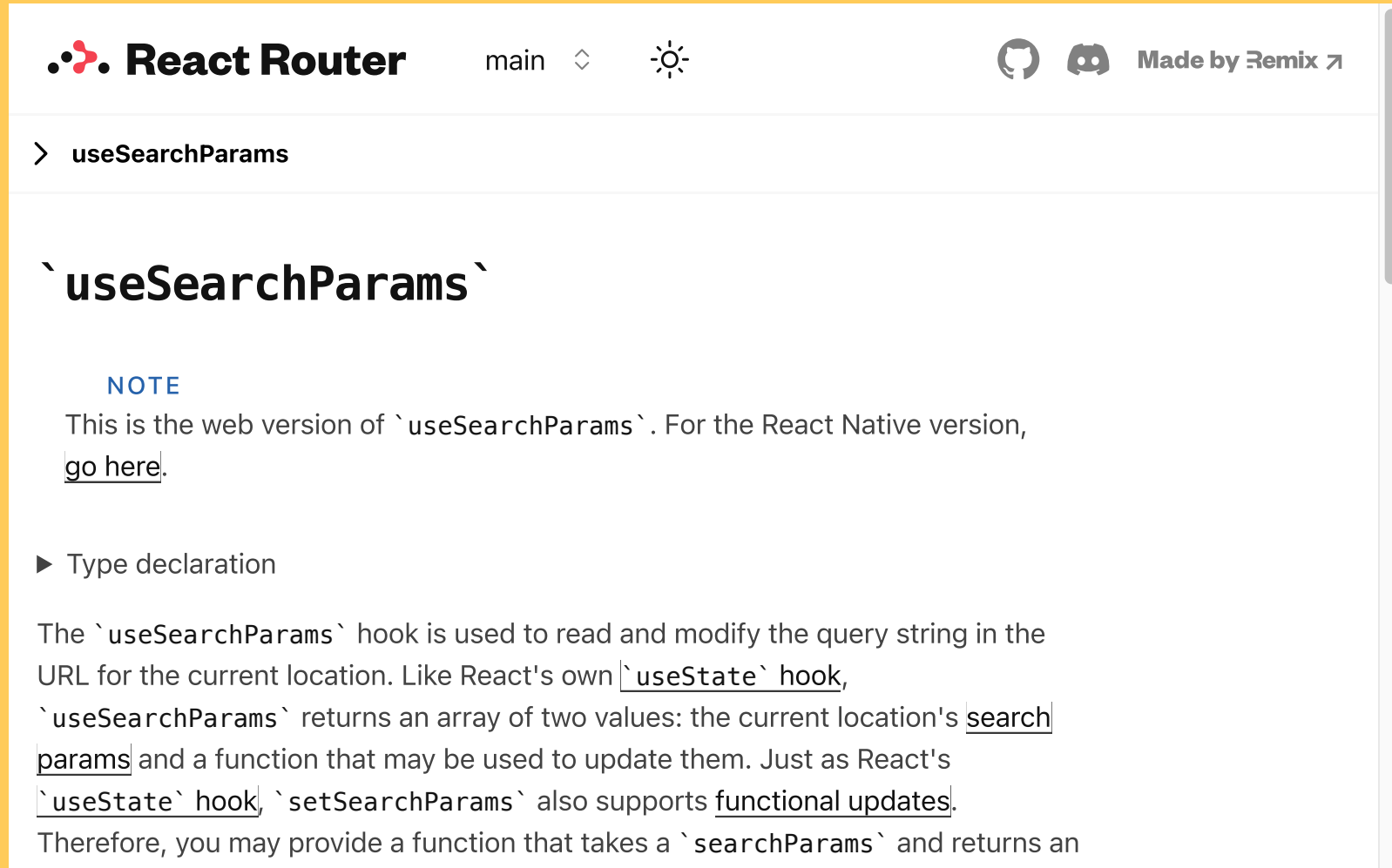
NAVIGATE

Переадресація

```
<route path="/lesson1" element="{<Lesson1">}">Home</route>  
      <route path="/lesson9" element="{<Navigate" to="/
```

USEMATCH, USEPARAMS

<https://reactrouter.com/en/main/hooks/use-search-params>



The screenshot shows the React Router documentation page for the `useSearchParams` hook. The page header includes the React Router logo, the text "main", a search icon, GitHub and Discord icons, and "Made by Remix". The breadcrumb navigation shows `useSearchParams`. The main heading is `useSearchParams``. A "NOTE" section states: "This is the web version of `useSearchParams``. For the React Native version, [go here](#)." A "Type declaration" section is partially visible. The main text explains that the `useSearchParams`` hook is used to read and modify the query string in the URL for the current location. It compares it to React's `useState`` hook, stating that `useSearchParams`` returns an array of two values: the current location's `search params` and a function to update them. It also notes that `setSearchParams`` supports `functional updates`. The text concludes with "Therefore, you may provide a function that takes a `searchParams`` and returns an

REACT.LAZY

```
const Header = lazy(() => import('./components/Header'));
```

REACT.SUSPENSE

```
<suspense fallback="{ 'Loading...'}" >  
  <header favcolor="blue"></header>  
</suspense>
```

WHO? WHO?

