Open Opened 3 weeks ago by Rubén Montero

Varias páginas con React



- Actualizaremos nuestro repositorio con git pull y comenzaremos a trabajar en la nueva carpeta react-dashboards/
- Instalaremos <u>react-router-dom</u>
- Implementaremos una versión sencilla de aplicación multipágina, con dos páginas web

Descripción

Recordemos que React es un framework para la creación de interfaces web basadas en componentes. La filosofía de React se limita a:

- 1. Servir al cliente un montón de JavaScript y un HTML <div></div> vacío
- 2. El JavaScript se ejecuta después de ser obtenido por el navegador, o en respuesta a interacciones del usuario.
- 3. Cuando esto sucede, modifica el DOM para que se visualice algo distinto a un sencillo <div></div>
- 4. El usuario disfrutará del nuevo contenido y de su rápida velocidad de respuesta

Estamos hablando de una SPA (single-page application). Una sola URL. Una sóla página.

Client-side routing con react-router-dom

Esta librería nos confiere los componentes necesarios para manejar navegación dentro la página web. Por debajo, usa <u>la librería history</u> y <u>React context</u> para implementar *navegación* sin renunciar a los beneficios de las SPAs.

Ya lo hemos puesto en práctica. ¡Pongámoslo de nuevo!



La tarea

En primer lugar, efectúa git pull de tu repositorio para asegurarte que estás al día.

Después, desde un terminal (cmd.exe), **posiciónate** (cd) en la carpeta react-dashboards/react-frontend/ ¹ y **ejecuta**:

```
npm install
```

Cuando termine, para instalar la librería react-router-dom, ejecuta:

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

A continuación, **crea** una carpeta screens/ dentro de src/ donde albergaremos el código de cada página. Dentro, **crea** dos carpetas separadas (main/y about/).

Crea dos nuevos ficheros en cada carpeta, así:

• src/screens/main/Main.js

```
const Main = () => {
    return <h2 data-cy='pageHeader'>
        Principal
        </h2>
}
export default Main;
```

• src/screens/about/About.js

```
const About = () => {
    return <h2 data-cy='pageHeader'>
        Acerca de
        </h2>
}
export default About;
```

https://raspi/francisco.gomez/dwec/issues/95

Ahora, **elimina** el contenido por defecto en App.js:

```
function App() {
 return (
     <div className="App">
       <header className="App-header">
         <img src={logo} className="App-logo" alt="logo" />
         >
          Edit <code>src/App.js</code> and save to reload.
         ≺a
           className="App-link"
          href="https://reactjs.org"
          target="_blank"
           rel="noopener noreferrer"
           Learn React
         </a>
       </header>
     </div>
 );
}
```

...y, en su lugar, **añade** un <BrowserRouter> y un <Routes> :

La parte fundamental: Definir las rutas

Para terminar, dentro del <Routes> </Routes> , añade estas dos rutas:

```
<Route path="/" element={<Main/>}></Route>
<Route path="/about" element={<About/>}></Route>
```

(Y, recuerda, los imports necesarios):

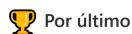
```
import { BrowserRouter, Routes, Route } from "react-router-dom";
import Main from './screens/main/Main';
import About from './screens/about/About';
```

Prueba a lanzar el servidor con npm run start

¿Ves correctamente las siguientes páginas?

- http://localhost:3000/
- http://localhost:3000/about

¡Enhorabuena! Has arrancado el proyecto con éxito.



Sube tus cambios al repositorio en un nuevo commit con un buen mensaje descriptivo.

No está de más verificar que tu commit se ha subido correctamente desde GitLab (https://raspi).

1. ¡Ojo! En este *sprint* tendremos dos carpetas principales: react-dashboards/react-frontend/ contiene el proyecto React y trabajaremos sobre ella. react-dashboards/django-backend/ contiene código que es parte del proyecto, pero del *backend*. No trabajaremos aquí, pero puedes echar un vistazo si te da curiosidad

https://raspi/francisco.gomez/dwec/issues/95



Rubén Montero @ruben.montero changed milestone to %Sprint 4 3 weeks ago



Francisco Gómez @francisco.gomez mentioned in commit 6f07e490 3 weeks ago