

Drag and Drop a los enlaces de DevTree

Librería: dnd kit

```
16:25:01 [vite] (client) hmr update /src/views/LinkTreeView.tsx, /src/index.css (x8)
PS C:\Users\rodri\devtree\frontend> npm i @dnd-kit/core @dnd-kit/sortable @dnd-kit/utilities

added 4 packages, and audited 320 packages in 18s

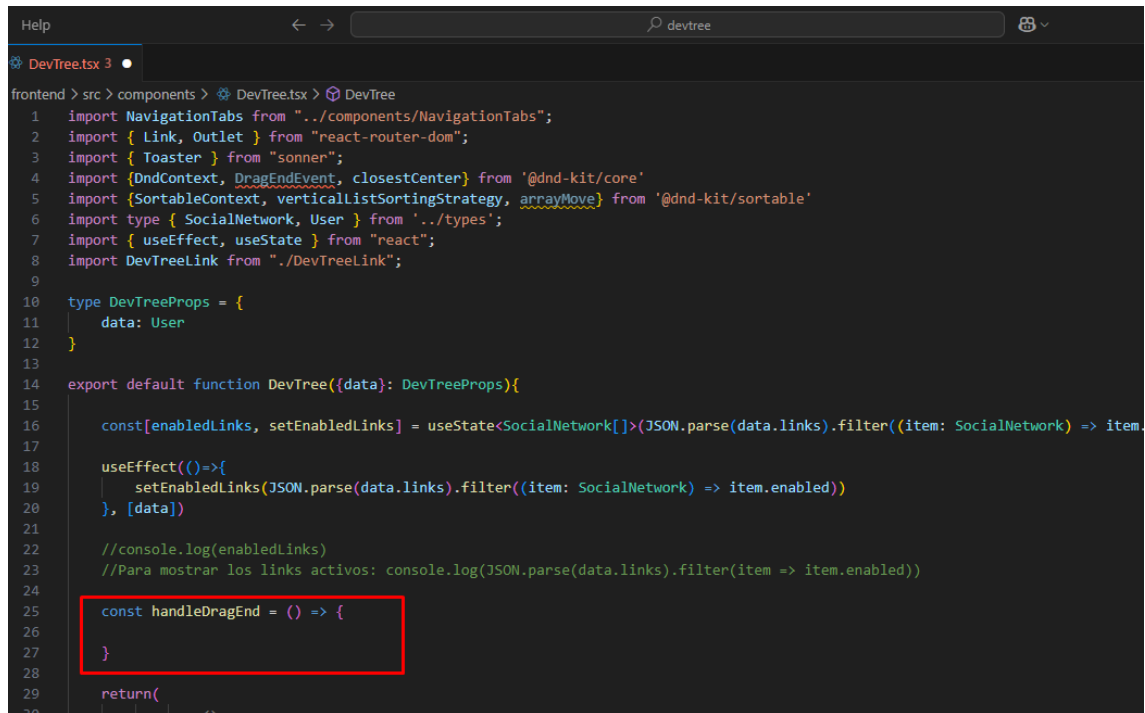
87 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\rodri\devtree\frontend> 
```

```
DevTree.tsx 3
frontend > src > components > DevTree.tsx > ...
1 import NavigationTabs from "../components/NavigationTabs";
2 import { Link, Outlet } from "react-router-dom";
3 import { Toaster } from "sonner";
4 import { DndContext, DragEndEvent, closestCenter } from "@dnd-kit/core";
5 import { SortableContext, verticalListSortingStrategy, arrayMove } from "@dnd-kit/sortable";
6 import type { SocialNetwork, User } from "../types";
7 import { useEffect, useState } from "react";
8 import DevTreeLink from "../DevTreeLink";
9
10 type DevTreeProps = {
11   data: User
12 }
13
14 export default function DevTree({data}: DevTreeProps){
15
16   const [enabledLinks, setEnabledLinks] = useState<SocialNetwork[]>(JSON.parse(data.links).filter((item: SocialNetwork) => item.
17
18   useEffect(()=>{
```

Añadiendo los Componentes Principales

```
DevTree.tsx 3
frontend > src > components > DevTree.tsx > DevTree
14 export default function DevTree({data}: DevTreeProps){
15
16   const [enabledLinks, setEnabledLinks] = useState<SocialNetwork[]>(JSON.parse(data.links).filter((item: SocialNetwork) => item.
17
18   useEffect(()=>{
19     // Cargar los datos de la API
20     fetch('http://localhost:3001/users')
21       .then(response => response.json())
22       .then(data => {
23         setEnabledLinks(data.links);
24       })
25       .catch(error => console.error('Error al cargar los datos:', error));
26   });
27
28   // Renderizar el árbol de enlaces
29   return (
30     <div className="w-full md:w-96 bg-slate-800 px-5 py-10 space-y-6">
31       <p className="text-4xl text-center text-white">{data.handle}</p>
32
33       {data.image && //si tenemos una imagen, la renderizamos, pero si no:
34         <img src={data.image} alt="Imagen Perfil" className="mx-auto max-w-[250px]" />
35       }
36
37       <p className="text-center text-lg font-black text-white">{data.description}</p>
38
39       <DndContext
40         collisionDetection={closestCenter}
41         onDragEnd={handleDragEnd}
42       >
43         <div className="mt-20 flex flex-col gap-5">
44           <SortableContext
45             items={enabledLinks}
46             strategy={verticalListSortingStrategy}
47           >
48             {enabledLinks.map(link => (
49               <DevTreeLink key={link.name} link={link} />
50             ))}
51           </SortableContext>
52         </div>
53       </DndContext>
54     </div>
55   );
56 }
```



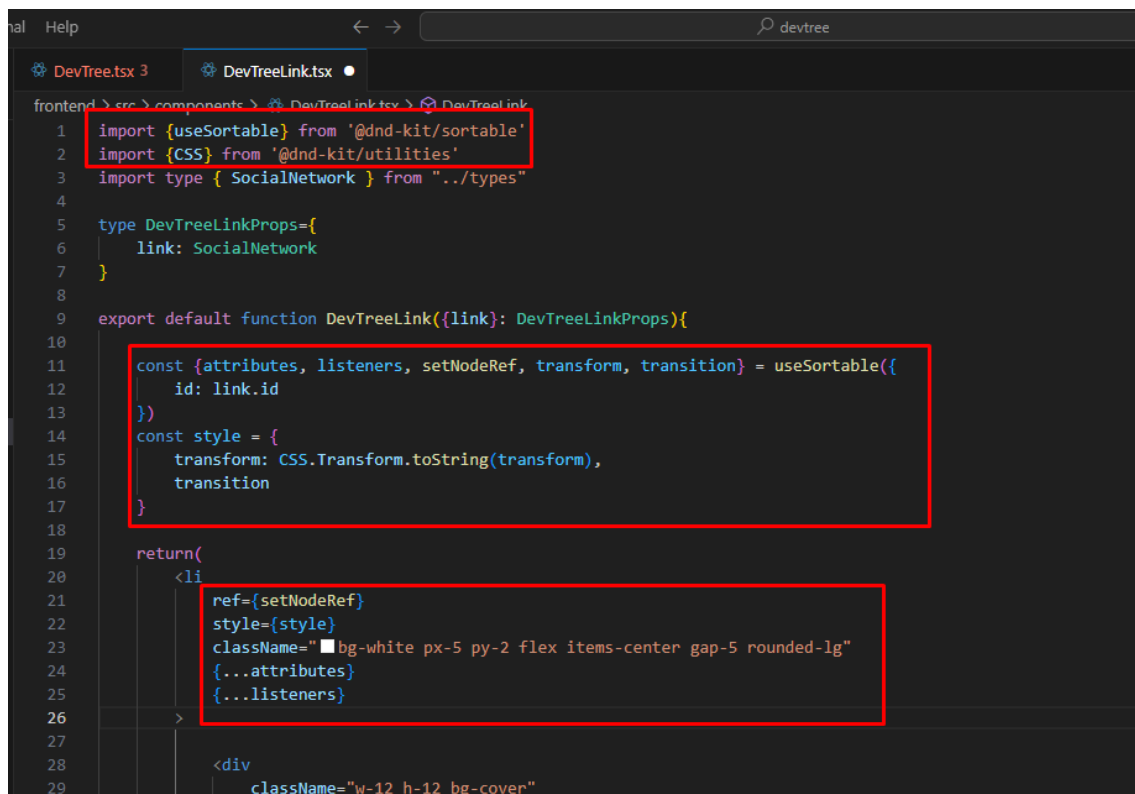
```

1 import NavigationTabs from "../components/NavigationTabs";
2 import { Link, Outlet } from "react-router-dom";
3 import { Toaster } from "sonner";
4 import { DndContext, DragEndEvent, closestCenter } from '@dnd-kit/core'
5 import { SortableContext, verticalListSortingStrategy, arrayMove } from '@dnd-kit/sortable'
6 import type { SocialNetwork, User } from '../types';
7 import { useEffect, useState } from "react";
8 import DevTreeLink from "../DevTreeLink";
9
10 type DevTreeProps = {
11   data: User
12 }
13
14 export default function DevTree({data}: DevTreeProps){
15
16   const[enabledLinks, setEnabledLinks] = useState<SocialNetwork[]>(JSON.parse(data.links).filter((item: SocialNetwork) => item.enabled))
17
18   useEffect(()=>{
19     setEnabledLinks(JSON.parse(data.links).filter((item: SocialNetwork) => item.enabled))
20   }, [data])
21
22   //console.log(enabledLinks)
23   //Para mostrar los links activos: console.log(JSON.parse(data.links).filter(item => item.enabled))
24
25   const handleDragEnd = () => {
26
27   }
28
29   return(
30

```

Ahora, trabajaremos en el código HTML para que tenga ese comportamiento de poder arrastrar y soltar.

Añadiendo la funcionalidad de Drag



```

1 import { useSortable } from '@dnd-kit/sortable'
2 import { CSS } from '@dnd-kit/utilities'
3 import type { SocialNetwork } from "../types"
4
5 type DevTreeLinkProps={
6   link: SocialNetwork
7 }
8
9 export default function DevTreeLink({link}: DevTreeLinkProps){
10
11   const {attributes, listeners, setNodeRef, transform, transition} = useSortable({
12     id: link.id
13   })
14   const style = {
15     transform: CSS.Transform.toString(transform),
16     transition
17   }
18
19   return(
20     <li
21       ref={setNodeRef}
22       style={style}
23       className="bg-white px-5 py-2 flex items-center gap-5 rounded-lg"
24       {...attributes}
25       {...listeners}
26     >
27
28     <div
29       className="w-12 h-12 bg-cover"

```



Ya lo podemos mover, pero, al intentar subir Instagram, se regresa a su lugar de origen porque el ID no se está actualizando y para ello usaremos arrayMove.

```

14 export default function DevTree({data}: DevTreeProps){
15   useEffect(()=>{
16     setEnabledLinks(JSON.parse(data.links).filter((item: SocialNetwork) => item.enabled))
17   }, [data])
18
19   //console.log(enabledLinks)
20   //Para mostrar los links activos: console.log(JSON.parse(data.links).filter(item => item.enabled))
21
22   const handleDragEnd = (e: DragEndEvent) => {
23     const {active, over} = e
24
25     if(over && over.id){
26       const prevIndex = enabledLinks.findIndex(link => link.id === active.id)
27       const newIndex = enabledLinks.findIndex(link => link.id === over.id)
28       const order = arrayMove(enabledLinks, prevIndex, newIndex)
29       setEnabledLinks(order)
30     }
31   }
32
33   return(
34     <div>
35       <div>
36         <div>
37           <div>
38             <div>

```

Ya podemos probar la funcionalidad en el panel. Sin embargo, si habilitamos una nueva red social, todo se desordena.

Actualizando el orden a soltar

```

15 export default function DevTree({data}: DevTreeProps){
16   useTreeClient()
17 }, [data])
21
22
23 const queryClient = useQueryClient()
24
25 //console.log(enabledLinks)
26 //Para mostrar los links activos: console.log(JSON.parse(data.links).filter(item => item.enabled))
27
28 const handleDragEnd = (e: DragEndEvent) => {
29
30   const {active, over} = e
31
32   if(over && over.id){
33     const prevIndex = enabledLinks.findIndex(link => link.id === active.id)
34     const newIndex = enabledLinks.findIndex(link => link.id === over.id)
35     const order = arrayMove(enabledLinks, prevIndex, newIndex)
36     setEnabledLinks(order)
37
38     const disabledLinks : SocialNetwork[] = JSON.parse(data.links).filter((item: SocialNetwork) => !item.enabled)
39
40     const links = order.concat(disabledLinks)
41
42     //setear el caché de usuario
43     queryClient.setQueryData(['user'], (prevData: User) => {
44       return{
45         ...prevData,
46         links: JSON.stringify(links)
47       }
48     })
49   }
50 }
51
52

```

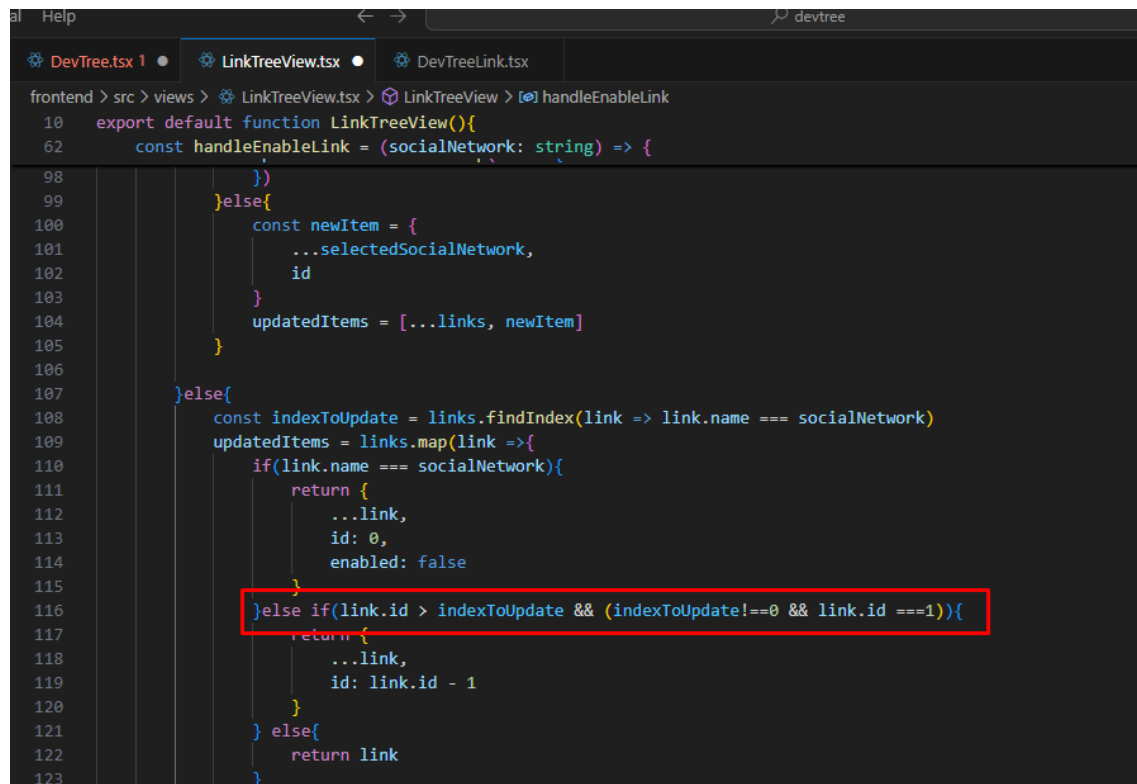
Le indicamos que cuando el usuario dé clic en Guardar Cambios, obtenga los datos del caché.

```

10 export default function LinkTreeView(){
11   const handleEnableLink = (socialNetwork: string) => {
12     queryClient.setQueryData(['user'], (prevData: User) => {
13       return{
14         ...prevData,
15         links: JSON.stringify(socialNetwork)
16       }
17     })
18   }
19
20   return(
21     <div className="space y-5">
22       {devTreeLinks.map(item=>{
23         <DevTreeInput
24           key={item.name}
25           //pasamos el item hacia el componente
26           item={item}
27           handleUrlChange={handleUrlChange}
28           handleEnableLink={handleEnableLink}
29         />
30       })}
31     <button
32       className="bg-cyan-400 p-2 text-lg w-full uppercase text-slate-600 rounded-lg font-bold"
33       onClick={() => mutate(queryClient.getQueryData(['user'])!)} //colocamos un callback para que espere por ese evento
34     > Guardar cambios</button>
35   </div>
36 )
37 }
38
39

```

Si deshabilitamos una red Social, tendrá id 0, y el Drag And Drop no permitirá arrastrar a quienes tengan id 0.



```

10 export default function LinkTreeView(){
62   const handleEnableLink = (socialNetwork: string) => {
98     }
99   }else{
100     const newItem = {
101       ...selectedSocialNetwork,
102       id
103     }
104     updatedItems = [...links, newItem]
105   }
106 }else{
107   const indexToUpdate = links.findIndex(link => link.name === socialNetwork)
108   updatedItems = links.map(link =>{
109     if(link.name === socialNetwork){
110       return {
111         ...link,
112         id: 0,
113         enabled: false
114       }
115     }else if(link.id > indexToUpdate && (indexToUpdate !== 0 && link.id === 1)){
116       return {
117         ...link,
118         id: link.id - 1
119       }
120     } else{
121       return link
122     }
123   }

```

Probamos y todo bien.

