fetch es una API nativa del navegador para realizar peticiones al servidor. No está soportada por todos los navegadores, pero existen *polyfills* para agregar esta funcionalidad. El *polyfill* más conocido es github/fetch.

`fetch` toma como primer parámetro la URL a la que queremos realizar la petición. El segundo es un objeto opcional para modificar parámetros de la petición como los *headers* o el método.

Vamos a actualizar el fichero src/containers/SearchContainer/SearchContainer.js para que realice las peticiones a la API de Github. El *endpoint* que utilizaremos es GET /search/repositories.

Sólo necesitamos modificar el método `onSubmit` para que realice la petición y lance las acciones de Redux.

```
/**
 * Este método actua como callback del evento onSubmit del formulario.
 * Recibe como parámetro el campo que debe de buscar.
 */
onSubmit = value => {
  // Lanzamos la accion!
  this.props.dispatch(startSearch(value));
  // Realizamos la petición a la API
  fetch(`https://api.github.com/search/repositories?q=${ value }`)
    .then(res => {
      return res.json();
    })
    .then(res => {
      this.props.dispatch(successSearch(res.items));

    })
    .catch(err => {
      // Mostramos el error por consola
      console.log(err);
    })
}
```

Con este cambio, nuestro componente ya realizará las peticiones al servidor.

Como ejercicio, podéis modificar el código de `src/containers/DetailsContainer/DetailsContainer.js` para que realice peticiones a la API. El endpoint que debéis utilizar es el de Releases. Recordad que `DetailsContainer` no necesita almacenar los datos en el estado, por lo que podéis almacenar los valores en el estado. La solución la tenéis en el repositorio: src/containers/DetailsContainer/DetailsContainer.js.