JUNIORDev DIGITALNA DALMACIJA

Komunikacija > Axios

Axios biblioteka

Osim standardnog *fetch API*-ja, jedna od najčešće korištenih biblioteka za slanje HTTP zahtjeva je Axios koja pruža neke dodatne mogućnosti i olakšava komunikaciju sa poslužiteljem. Za razliku od *fecth API*-ja koji je dio ugrađenih WEB API metoda, *Axios* je prije korištenja potrebno instalirati u našu aplikaciju.

Instalacija

Instalaciju ponovno radimo pomoću ugrađenog Terminala u VS Code pri čemu pazimo na trenutnu lokaciju (mora odgovarati direktoriju aplikacije). *Axios* instaliramo pomoću naredbe:

```
npm install axios
```

Nakon instalacije možemo koristiti *axios* funkcionalnosti u našim komponentama, naravno uz uključivanje same biblioteke:

```
import axios from "axios";
```

Dohvat podataka

Prisjetimo se kako smo u prethodnim primjerima koristili *fetch* naredbu (bez automatskog dohvaćanja i *useEffect hook*-a):

App.jsx

```
import { useState } from "react";

function App() {
  const [podatak, postaviPodatak] = useState({
    body: "",
    id: null,
    title: "",
    userId: null,
  });

  function dohvatiPodatke() {
    fetch("https://jsonplaceholder.typicode.com/posts/1")
      .then(res => res.json())
      .then(data => postaviPodatak(data));
  }

  return (
    <div className='App'>
      <h1>Dohvat podataka</h1>
      <button onClick={dohvatiPodatke}>Dohvati podatke</button>
      <div>
        <h3>{podatak.title}</h3>
        <p>{podatak.body}</p>
      </div>
    </div>
  );
}
```

Za početak ćemo uključiti *axios* sa naredbom `import axios from "axios"` te pogledati u kojem obliku dobijemo odgovor. Za dohvat podataka umjesto `fetch()` koristimo metodu `axios.get()`

```
  function dohvatiPodatke() {
    axios.get("https://jsonplaceholder.typicode.com/posts/1")
      .then(res => console.log(res))
  }
```

Ako pogledamo u konzolu, vidjeti ćemo da je, za razliku od *fetch*, *axios* automatski parsirao tijelo (*body*) odgovora te se u svojstvu ***data*** nalaze naši traženi podaci. Kada to znamo,

možemo odmah podatke spremiti u varijablu stanja.

```
function dohvatiPodatke() {
  axios.get("https://jsonplaceholder.typicode.com/posts/1")
    .then(res => postaviPodatak(res.data))
}
```

Već na ovom primjeru vidimo neke od pogodnosti *axios* biblioteke - nije bilo potrebe za ulančavanjem `then()` metoda niti parsiranjem tijela odgovora.

Upravljanje greškama

Pogledajmo još jedan primjer prednosti korištenja *axios*-a. Vratiti ćemo se na *fetch* metodu i namjerno poslati krivi URL. Vidjeli smo kod *Promise* objekta da možemo upravljati pogreškama sa `.catch()` metodom. Ovaj kôd bi prema tome trebao ispisati poruku sa pogreškom jer ne postoji podatak sa ID vrijednosti 101:

```
function dohvatiPodatke() {
  fetch("https://jsonplaceholder.typicode.com/posts/101")
    .then(res => {res.json()})
    .then(data => postaviPodatak(data))
    .catch(err => alert(err))
}
```

Kada pozovemo ovu funkciju ništa se ne događa (preglednik će ispisati pogrešku na konzoli ali to nije rezultat aplikacije). Problem je što *fetch* statusne kôdove `4xx - 5xx` ne shvaća kao pogreške jer to strogo gledano i nisu - poslužitelj nam je vratio odgovor, bez obzira što je sadržaj tog odgovora neki oblik poruke da zahtjev nije bio dobar (4xx) ili da poslužitelj nije mogao obraditi zahtjev (5xx).

Ako ponovno ispišemo inicijalni odgovor u konzolu (prije parsiranja) vidjeti ćemo da nam *fetch* odgovor sadrži dva korisna svojstva - ***ok*** koji je *boolean* vrijednost i ***status*** koji je cjelobrojna vrijednost. Pomoću ovih svojstava možemo ugraditi dodatne provjere i upravljanje greškama ali to doprinosi

S druge strane, *axios* automatski prepoznaje neispravne zahtjeve kao pogreške, te možemo vrlo jednostavno upravljati pogreškama kao što je prikazano na primjeru ispod:

```
function dohvatiPodatke() {
  axios.get("https://jsonplaceholder.typicode.com/posts/101")
    .then(res => postaviPodatak(res.data))
    .catch(err => alert(err))
}
```

Ovaj put će nam se prilikom slanja zahtjeva ispisati poruka sa pogreškom na *alert* prozoru. *Axios* sadrži još neke dodatne funkcionalnosti koje će nam biti korisne kada budemo slali druge tipove zahtjeva pa ćemo zbog toga u svim daljnjim primjerima sa slanjima zahtjeva koristiti *axios* biblioteku.

Last updated on March 28, 2023

‹ Slanje zahtjeva Niz podataka ›