10-AXIOS-HOOKS

#01. 프로젝트 생성 및 초기화

1) 프로젝트 생성

> 프로젝트 이름은 영어 소문자만 사용 가능함.

```
yarn create react-app 10-axios-hooks
```

2) 프로젝트 생성 후 기초작업

프로젝트 초기화

대상	작업 내용
src폴더	App.css, App.test.js, index.css, logo.svg, setupTests.js, reportWebVitals.js 삭제
App.js	App.css와 logo.svg에 대한 참조(import) 구문 제거
index.js	index.css와 reportWebVitals.js에 대한 참조(import) 구문 제거 맨 마지막 행에 있는 `reportWebVitals()` 부분 삭제

3) 추가 패키지 설치

패키지 이름	설명
react-router-dom	SPA앱을 만들 때 사용. URL에 따라 실행할 Javascript를 분기한다.
react-helmet-async	`<head>`태그 내의 SEO관련 태그를 설정한다.
sass	sass와 scss 컴파일 기능 제공
styled-components	styled component 지원
styled-components-breakpoints	styled component에서 media query를 쉽게 사용할 수 있게 한다.
dayjs	날짜 처리 기능 제공 (리액트와 직접적인 연관은 없음)
axios	Ajax 요청 라이브러리
react-loader-spinner	로딩바(spinner) 컴포넌트
axios-hooks	Axios 라이브러리 사용을 간소화 시켜 주는 hook

```
yarn add react-router-dom react-helmet-async sass styled-components
styled-components-breakpoints dayjs axios react-loader-spinner axios-hooks
```

4) Router 적용

index.js

import 구문 추가

```
import { BrowserRouter } from 'react-router-dom';
```

index.js 파일에서 `<App />`을 `<BrowserRouter><App /></BrowserRouter>`로 변경

App.js

```
import { NavLink, Routes, Route } from "react-router-dom";
```

혹은

```
import { Link, Routes, Route } from "react-router-dom";
```

5) 프로젝트 실행

프로젝트를 VSCode로 열고, `Ctrl` + `~`를 눌러 터미널 실행

```
yarn start
```

#02. `useAxios()` 사용하기

1) 기본 사용 방법

```
const [{ data, loading, error }, refetch] = useAxios(url);
```

기본 GET 방식 요청

- data: 응답결과로 수신된 json 결과
- loading: 로딩여부를 나타내는 boolean값
- error : 에러가 존재할 경우 에러 정보를 내장하고 있는 객체
- refetch : 요청을 다시 보내고자 할 경우 사용할 함수
- url : 요청 URL (path 파라미터 혹은 querystring을 포함하고 있어야 함)

2) CRUD에 따른 요청 방법

GET

```
const [{ data, loading, error }, refetch] = useAxios({
    url: '요청URL',
    method: 'GET',
    params: { key: value, key: value }
});
```

params에 설정한 정보는 querystring으로 전송됨

POST, PUT, DELETE

```
const [{ data, loading, error }, refetch] = useAxios({
    url: '요청URL',
    method: 'POST|PUT|DELETE',
    data: { key: value, key: value }
});
```

3) 메뉴얼 요청(수동 요청)

hook 기능을 사용하면 컴포넌트가 마운트됨과 동시에 ajax요청이 전송된다.

입력, 수정, 삭제, 검색 등의 기능은 사용자 이벤트가 발생했을 때 요청을 전송해야 하기 때문에 자동 전송되게 하지 않고 수동으로 전송할 수 있는 메뉴얼 기능을 사용해야 한다.

메뉴얼 요청 설정

`{ manual: true }`를 추가한다.

```
const [{ data, loading, error }, refetch] = useAxios({
        ... url, method, params, data 등 설정 ...
    }, { manual: true });
```

설정된 그대로 메뉴얼 요청 전송하기

`refetch()` 함수를 사용한다.

```
refetch();
```

GET 방식일 경우 요청 파라미터를 변경하여 전송하기

`refetch()` 함수에 전송할 parameter를 설정한다.

```
refetch({
    params: {key:value, key: value}
});
```

POST,PUT,DELETE 방식일 경우 요청 파라미터를 변경하여 전송하기

```
refetch({
    data: {key:value, key: value}
});
```

#03. 타이타닉 데이터

타이타닉(Titanic)은 영국의 화이트 스타 라인이 운영한 북대서양 횡단 여객선이다.

1912년 4월 10일 영국의 사우샘프턴을 떠나 미국의 뉴욕으로 향하던 첫 항해 중에 4월 15일 빙산과 충돌하여 침몰하였다. 배에는 승객들을 태울 충분한 구명보트가 없었고, 타이타닉의 침몰로 2,224명의 승객 중 1,502명이 사망하였다.

아래의 URL을 통해 타이타닉 사건때 배에 있었던 승객들 명단을 JSON으로 조회할 수 있다.

```
http://localhost:3001/tatanic
```

응답 데이터

변수명	설명
id	승객번호
survived	생존여부. true=생존, false=사망
pclass	1 = 1등석, 2 = 2등석, 3 = 3등석
name	승객이름
sex	male = 남성, female = 여성
age	나이
sibsp	타이타닉 호에 동승한 자매 / 배우자의 수
parch	타이타닉 호에 동승한 부모 / 자식의 수
ticket	티켓 번호
fare	승객 요금
cabin	방 호수
embarked	탑승지, C = 셰르부르, Q = 퀸즈타운, S = 사우샘프턴