CSR & SSR

> CSR : Client Side Rendering
>
> SSR : Server Side Rendering

CSR에는 모바일 시대에 들어서 SPA가 등장했다.

SPA(Single Page Applictaion)

> 최초 한 번 페이지 전체를 로딩한 뒤, 데이터만 변경하여 사용할 수 있는 애플리케이션

SPA는 기본적으로 페이지 로드가 없고, 모든 페이지가 단순히 Html5 History에 의해 렌더링된다.

기존의 전통적 방법인 SSR 방식에는 성능 문제가 있었다.

요즘 웹에서 제공되는 정보가 워낙 많다. 요청할 때마다 새로고침이 일어나면서 페이지를 로딩할 때마다 서버로부터 리소스를 전달받아 해석하고, 화면에 렌더링하는 방식인 SSR은 데이터가 많을 수록 성능문제가 발생했다.

```
현재 주소에서 동일한 주소를 가리키는 버튼을 눌렀을 때,
설정페이지에서 필요한 데이터를 다시 가져올 수 없다.
```

이는, 인터랙션이 많은 환경에서 비효율적이다. 렌더링을 서버쪽에서 진행하면 그만큼 서버 자원이 많이 사용되기 때문에 불필요한 트래픽이 낭비된다.

CSR 방식은 사용자의 행동에 따라 필요한 부분만 다시 읽어온다. 따라서 서버 측에서 렌더링하여 전체 페이지를 다시 읽어들이는 것보다 빠른 인터렉션을 기대할 수 있다. 서버는 단지 JSON파일만 보내주고, HTML을 그리는 역할은 자바스크립트를 통해 클라이언트 측에서 수행하는 방식이다.

뷰 렌더링을 유저의 브라우저가 담당하고, 먼저 웹앱을 브라우저에게 로드한 다음 필요한 데이터만 전달받아 보여주는 CSR은 트래픽을 감소시키고, 사용자에게 더 나은 경험을 제공할 수 있도록 도와준다.

CSR 장단점

- **장점**

- 트래픽 감소

  > 필요한 데이터만 받는다

- 사용자 경험

  > 새로고침이 발생하지 않음. 사용자가 네이티브 앱과 같은 경험을 할 수 있음

- **단점**

  - 검색 엔진

    > 크롬에서 리액트로 만든 웹앱 소스를 확인하면 내용이 비어있음. 이처럼 검색엔진 크롤러가 데이터 수집에 어려움이 있을 가능성 존재
    >
    > 구글 검색엔진은 자바스크립트 엔진이 내장되어있지만, 네이버나 다음 등 검색엔진은 크롤링에 어려움이 있어 SSR을 따로 구현해야하는 번거로움 존재

SSR 장단점

- **장점**

  - 검색엔진 최적화
  - 초기로딩 성능개선

    > 첫 렌더링된 HTML을 클라이언트에서 전달해주기 때문에 초기로딩속도를 많이 줄여줌

- **단점**

  - 프로젝트 복잡도

    > 라우터 사용하다보면 복잡도가 높아질 수 있음

  - 성능 악화 가능성

[참고 자료]

- 링크