리액트 기본 2

태그

CSR (Client-Side Rendering)과 SSR (Server-Side Rendering)의 차이점

1. Client-Side Rendering (CSR)

- **개념**: 브라우저가 클라이언트 측에서 자바스크립트를 실행해 콘텐츠를 렌더링합니다. 일반적으로 초기 페이지 로드는 빈 HTML 파일을 가져오고, 이후 브라우저가 자바스크립트 파일을 받아 실행하면서 콘텐츠가 렌더링됩니다.
- **장점**:
  - **인터랙티브한 사용자 경험**: CSR은 페이지 내비게이션 시 전체 페이지를 다시 로드하지 않고 콘텐츠가 동적으로 변경됩니다.
  - **개발 환경의 유연성**: 단일 페이지 애플리케이션(SPA)에서 상태 관리와 UI 업데이트를 더 쉽게 구현할 수 있습니다.
- **단점**:
  - **SEO 제한**: 초기 로드 시 검색 엔진 크롤러가 빈 HTML을 볼 수 있어 SEO 최적화가 어려울 수 있습니다.
  - **첫 로드 시간**: 초기 로드에 필요한 자바스크립트 파일이 커질 경우, 초기 페이지 로드 속도가 느려질 수 있습니다.

2. Server-Side Rendering (SSR)

- **개념**: 서버에서 HTML 페이지를 미리 렌더링하여 클라이언트에 전송합니다. 브라우저는 완전히 렌더링된 HTML 페이지를 받고, 그 후에 자바스크립트가 실행됩니다.
- **장점**:
  - **빠른 초기 로드**: 초기 콘텐츠가 서버에서 렌더링되므로, 클라이언트는 즉시 완성된 페이지를 볼 수 있습니다.
  - **SEO 최적화**: 검색 엔진 크롤러가 완전히 렌더링된 HTML을 수집할 수 있어, SEO 친화적입니다.
- **단점**:
  - **서버 부하**: 서버에서 렌더링하는 과정이 반복되면, 서버의 리소스 사용량이 증가할 수 있습니다.

    - **개발 복잡성 증가**: 서버와 클라이언트의 동기화를 유지하는 것이 어렵고, 개발 난이도가 높아질 수 있습니다.

React의 구조 및 장점

1. React의 구조

- **컴포넌트 기반 아키텍처**: React 애플리케이션은 재사용 가능한 컴포넌트들로 구성됩니다. 각 컴포넌트는 독립적으로 작동하며, 특정 UI와 관련된 로직을 캡슐화합니다.
- **Virtual DOM**: React는 실제 DOM을 직접 조작하지 않고, 가상 DOM을 사용하여 변경사항을 빠르게 계산하고, 필요한 부분만 실제 DOM에 업데이트합니다. 이를 통해 성능이 최적화됩니다.
- **단방향 데이터 바인딩**: React는 부모 컴포넌트에서 자식 컴포넌트로 데이터를 전달하는 단방향 데이터 흐름을 채택해 코드의 예측 가능성과 유지보수를 높입니다.

2. React의 장점

- **재사용성**: 컴포넌트를 사용하여 반복적인 코드 작성을 줄일 수 있습니다. 이를 통해 코드의 일관성을 유지하고 유지보수를 쉽게 할 수 있습니다.
- **효율적인 렌더링**: Virtual DOM 덕분에 DOM 조작이 최소화되어 빠른 UI 업데이트가 가능합니다.
- **풍부한 생태계**: React는 커뮤니티가 크고, 다양한 라이브러리와 도구들이 있어 확장성과 개발 속도가 뛰어납니다.
- **JSX 사용**: JSX를 통해 HTML과 JavaScript를 한 파일에 혼합하여 사용할 수 있습니다. 이는 코드의 가독성을 높이고 UI 코드를 직관적으로 작성할 수 있게 합니다.
- **강력한 상태 관리**: `useState`, `useReducer`와 같은 훅(Hook)을 통해 컴포넌트 상태를 쉽게 관리할 수 있습니다. 추가로 Redux나 MobX 같은 외부 상태 관리 라이브러리를 이용하면 더 큰 애플리케이션의 상태를 중앙 집중식으로 관리할 수 있습니다.
- **서버 사이드 렌더링 지원**: Next.js 같은 프레임워크와 결합하여 SSR을 구현할 수 있어 SEO와 초기 로딩 성능을 개선할 수 있습니다.

React는 유연성과 성능을 제공하면서도 다양한 프로젝트에서 사용하기 적합한 장점들이 많습니다. 단순한 웹 애플리케이션부터 대규모 애플리케이션까지 효율적으로 확장할 수 있는 구조를 제공합니다.