08. React에서 Next.js로

지금까지 우리는 React로 어떻게 시작할 수 있는지를 배워왔습니다.
최종 코드는 다음과 같았습니다.
(처음 시작하는 경우라면, 이 코드를 `index.html` 파일에 복사해 넣으면 됩니다.)

`index.html`

```
<html>
  <body>
    <div id="app"></div>

    <script src="https://unpkg.com/react@18/umd/react.development.js">
</script>
    <script src="https://unpkg.com/react-dom@18/umd/react-
dom.development.js"></script>
    <script src="https://unpkg.com/@babel/standalone/babel.min.js">
</script>

    <script type="text/jsx">
      const app = document.getElementById("app")

      function Header({ title }) {
        return <h1>{title ? title : "Default title"}</h1>
      }

      function HomePage() {
        const names = ["Ada Lovelace", "Grace Hopper", "Margaret
Hamilton"]

        const [likes, setLikes] = React.useState(0)

        function handleClick() {
          setLikes(likes + 1)
        }

        return (
          <div>
            <Header title="Develop. Preview. Ship." />
            <ul>
              {names.map((name) => (
                <li key={name}>{name}</li>
              ))}
            </ul>

            <button onClick={handleClick}>Like ({likes})</button>
```

```
          </div>
        )
      }

      const root = ReactDOM.createRoot(app);
      root.render(<HomePage />);
    </script>
  </body>
</html>
```

정리: 지금까지 배운 것

이전 챕터들에서는 React의 세 가지 핵심 개념을 소개했습니다:

- **컴포넌트(Components)**
- **프롭스(Props)**
- **상태(State)**

이 기본기를 갖추는 것은 React 애플리케이션을 구축하는 데 매우 중요합니다.

React를 배우는 가장 좋은 방법은 실제로 무언가를 만들어보는 것입니다.
지금까지 배운 내용만으로도 `<script>` 태그를 사용해 기존 웹사이트에 작은 컴포넌트를 추가하는 식으로 React를 점진적으로 도입할 수 있습니다.

하지만 많은 개발자들은 React가 제공하는 뛰어난 사용자 경험과 개발 경험 덕분에, 전체 프론트엔드 애플리케이션을 React로 작성하기로 결심하기도 합니다.

React → Next.js

React는 UI를 구축하는 데 매우 뛰어나지만,
UI를 **완전한, 확장 가능한 애플리케이션**으로 발전시키는 것은 별도의 작업이 필요합니다.

또한 최근 React에는 **Server Components**와 **Client Components** 같은 새로운 기능들이 추가되었는데,
이 기능들은 단독 React만으로는 다루기 어려워서 프레임워크의 도움이 필요합니다.

좋은 소식

Next.js는 이러한 설정과 구성을 대부분 자동으로 처리해줍니다.
또한 React 애플리케이션을 구축하는 데 유용한 추가 기능들도 제공합니다.

앞으로의 계획

이제 예제를 **React**에서 **Next.js**로 **마이그레이션**(이전)해 볼 것입니다.

또한:

- Next.js가 어떻게 동작하는지
- **Server Component**와 **Client Component**의 차이

에 대해서도 소개할 예정입니다.