Static Generation vs Server-side Rendering

Next.js

https://github.com/jio-ping

Next js의 pre-rendering

By default, Next.js **pre-renders** every page. This means that Next.js generates HTML for each page in advance, instead of having it all done by client-side JavaScript. Pre-rendering can result in better performance and SEO.

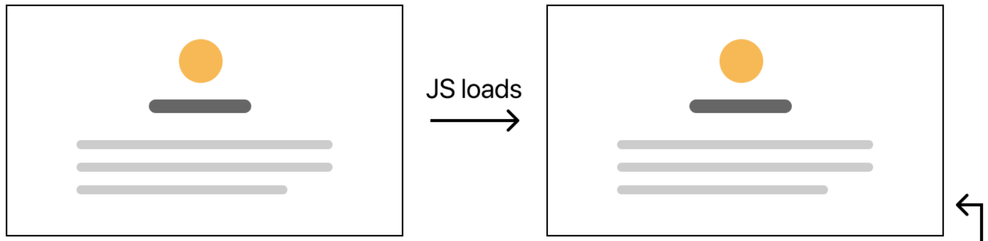
기본적으로 Next.js는 모든 페이지를 사전 렌더링합니다 .
즉, Next.js는 모든 것을 클라이언트 측 JavaScript로 처리하는 대신 각 페이지에 대한 HTML을 미리 생성합니다.
사전 렌더링은 더 나은 성능과 SEO를 가져올 수 있습니다.

pre-rendering vs No pre-rendering

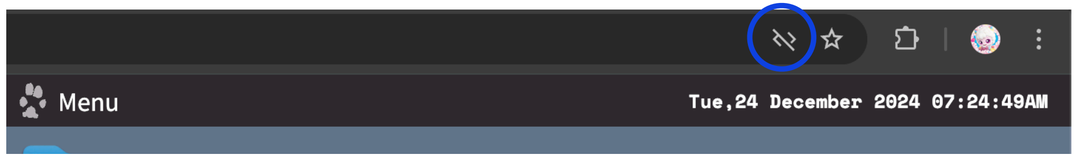
Pre-rendering (Using Next.js)

Initial Load:
Pre-rendered HTML is displayed

Hydration: React components are initialized and App becomes interactive



If your app has interactive components like `<Link />`, they'll be active after JS loads

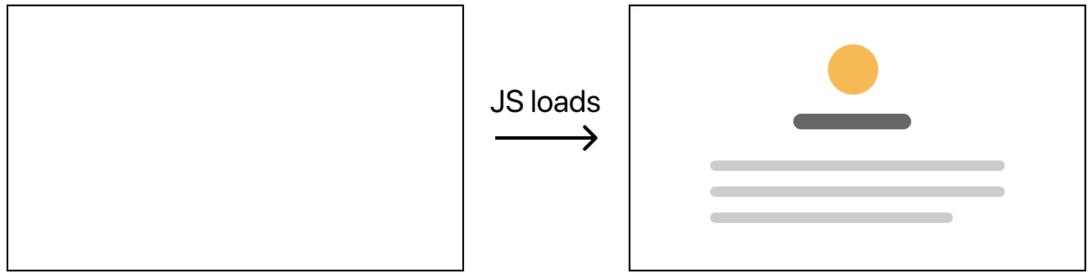


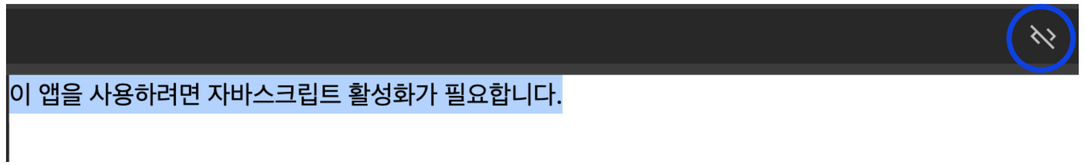
Static HTML으로 사전 렌더링 했기 때문에 javascript 없이도 UI를 확인

No Pre-rendering (Plain React.js app)

Initial Load:
App is not rendered

Hydration: React components are initialized and App becomes interactive





CSR 방식을 사용해 javascript 없이는 빈 화면이 나타남

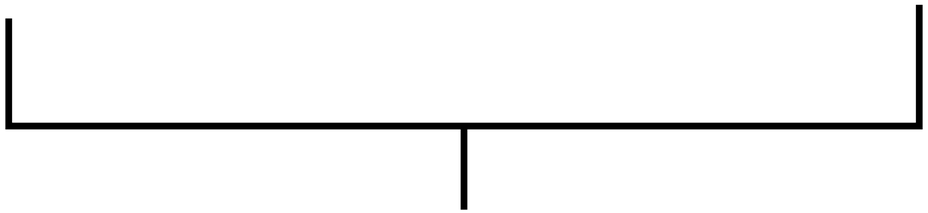
pre-rendering Mechanism

Static Site Generation

: 빌드시점에 생성된 정적 HTML 제공

Server Side Rendering

: 요청 시 서버에서 HTML 생성



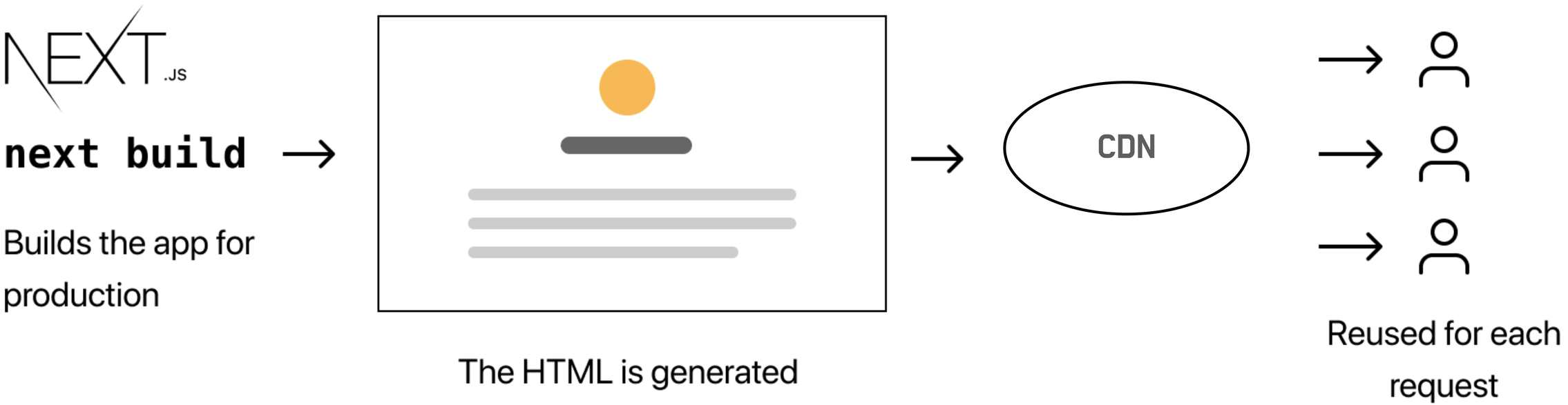
Hybrid App

각 페이지에 사용할 사전 렌더링 폼을 선택할 수 있음

Example)
Page1 : Static Generate
Page2 : Server Side Rendering
Page3 : Static Generate

Static Site Generation

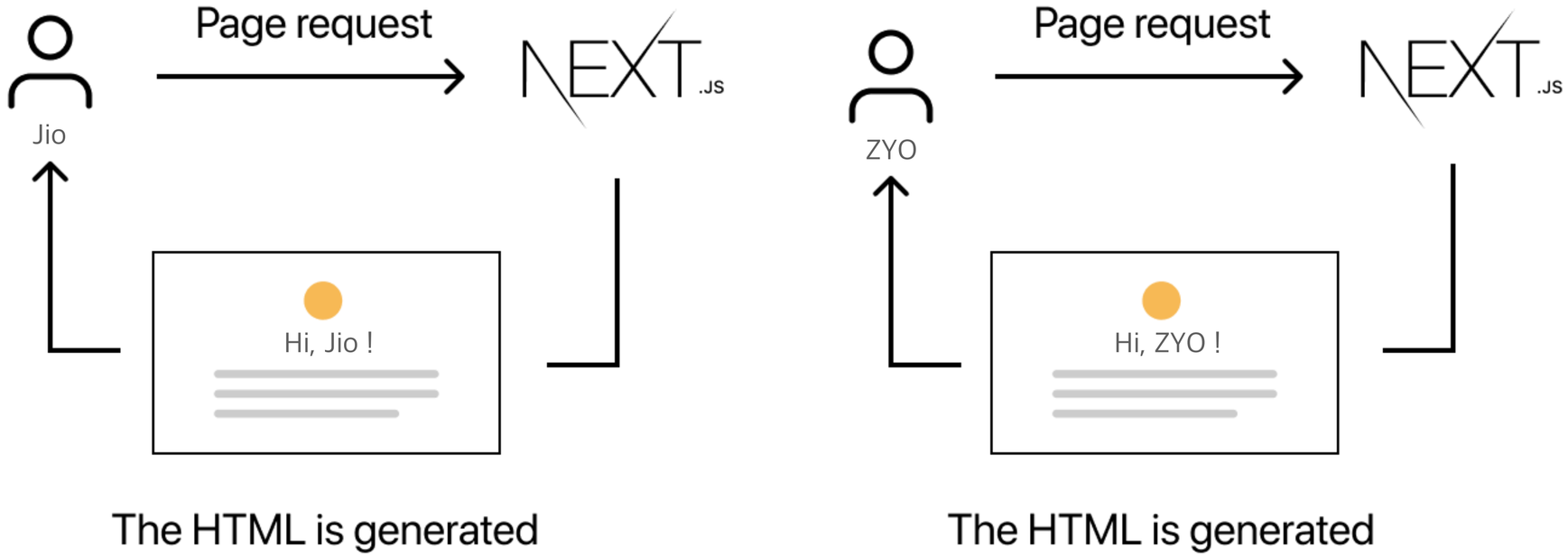
: 빌드시점에 생성된 정적 HTML 제공



빌드시, 데이터가 있는 경우 getStaticProps로 데이터를 가져와 HTML 파일을 생성
배포 후 정적 HTML 파일을 CDN에 저장
클라이언트의 각 요청에서 재 사용됨
JavaScript 파일이 로드 되면 페이지 상호작용 가능

Server Side Rendering(SSR)

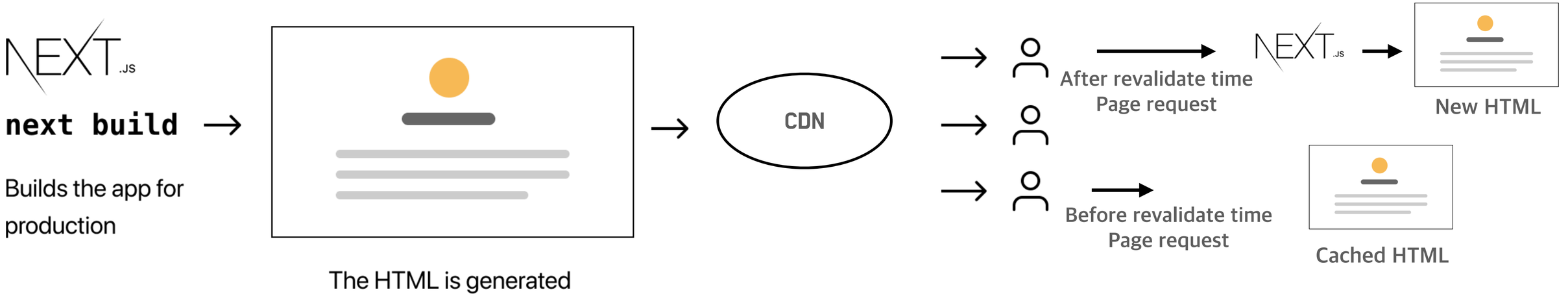
: 요청 시 서버에서 HTML 생성, Dynamic Rendering 이라고 부르기도 함



클라이언트가 요청시 getServerSideProps 메소드를 통해
서버에서 데이터를 가져와 HTML 생성 후 응답

Incremental Static Regeneration (ISR)

: SSG와 SSR의 혼합 방식. 빌드된 정적 페이지를 일정 간격으로 갱신



빌드 시 정적 HTML 생성 후 CDN에 저장

CDN은 초기 요청 시 정적 HTML 반환

Revalidate 시간 이후 사용자의 요청 시 server에서 새 HTML 생성 후 렌더링 & 캐시 교체

Next.js Rendering Methods Comparison

	Static Generation	Server Side Rendering	Incremental Static Regeneration
렌더링 시점	빌드시점	요청시점	빌드시점 + 설정된 주기
데이터 업데이트	재 배포 필요	실시간	Revalidate 시간 이후 요청시 업데이트
성능	가장빠름	상대적으로 느림	빠름
사용 케이스	SEO가 중요한 마케팅 페이지 자주 변경 되지 않는 블로그	실시간 대시보드 개인화된 피드/추천 인증이 필요한 페이지	주기적으로 업데이트 되는 e커머스 소셜 미디어 피드 주식시세 페이지

References

- nextjs.org[https://nextjs.org/learn-pages-router/basics/data-fetching/pre-rendering]
- next.js.org[https://nextjs.org/docs/pages/building-your-application/rendering/server-side-rendering]
- next.js.org [https://nextjs.org/docs/pages/building-your-application/data-fetching/incremental-static-regeneration]