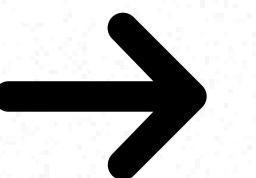
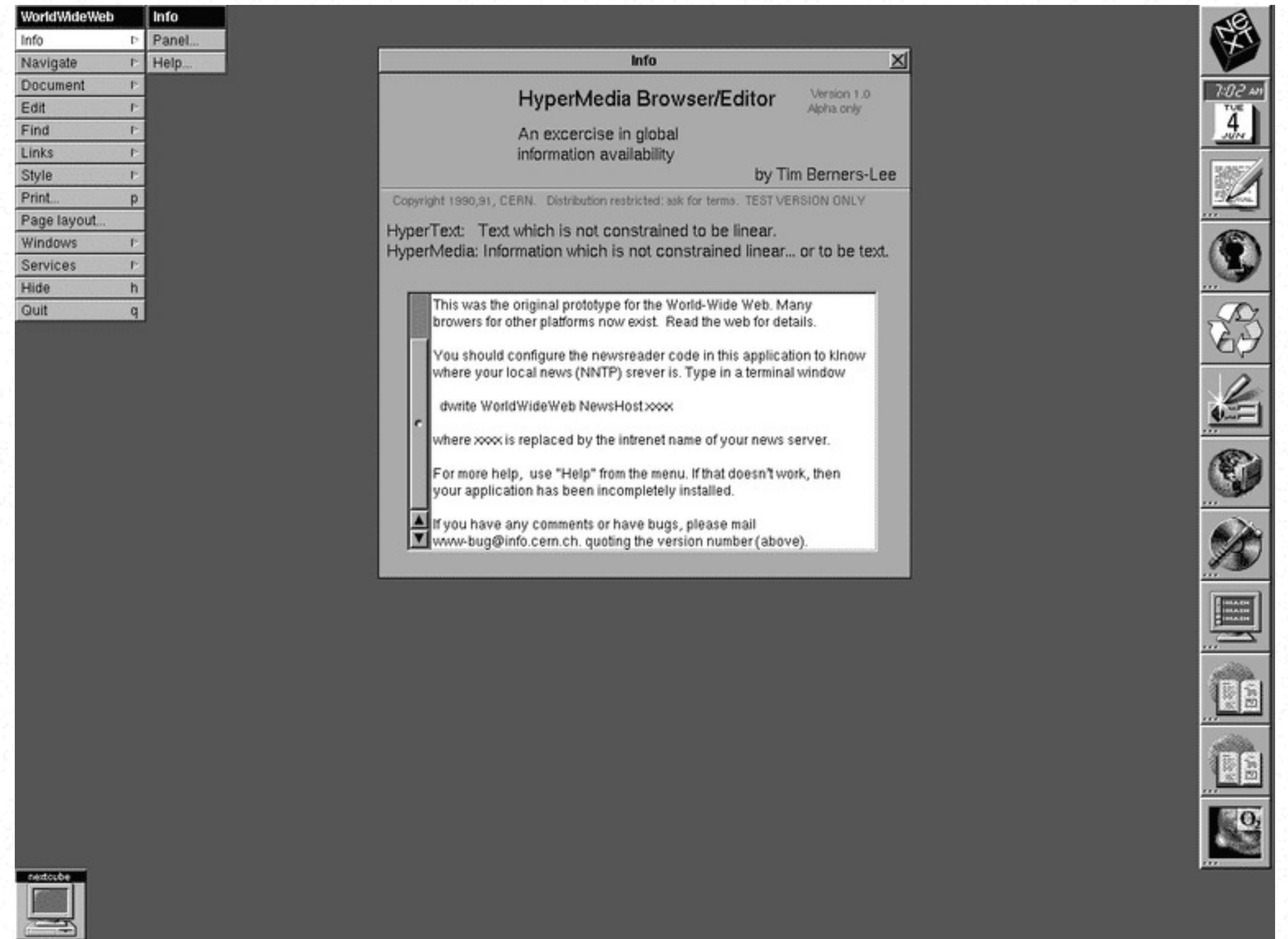


CSR & SSR

여민수

□ SPA와 MPA



이러한 전통적인 웹은 MPA 방식!

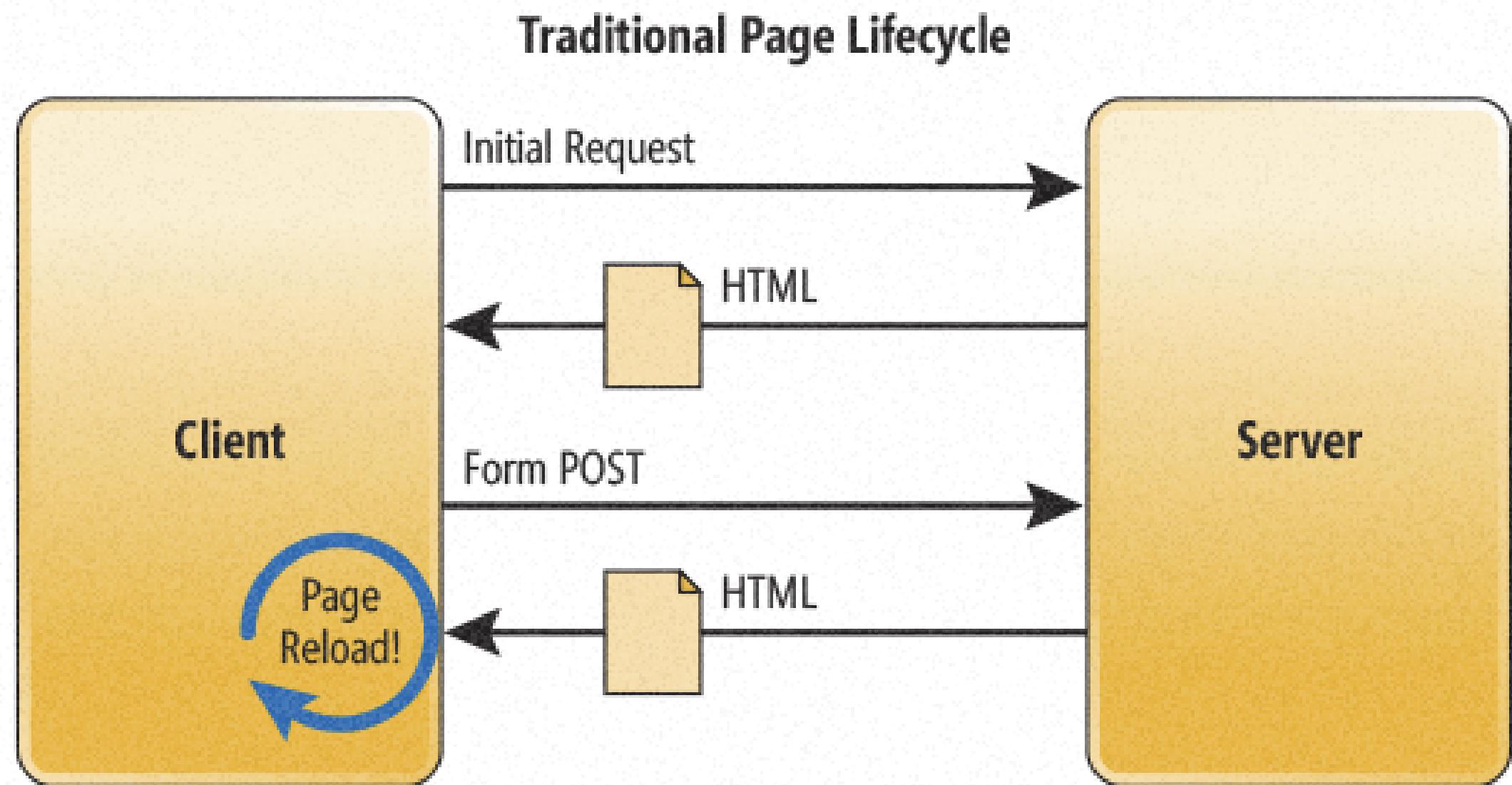
초창기 웹은 자료 공유 목적을 위한
텍스트 중심의 단순 문서



여러 개의 페이지로 구성된 애플리케이션

새로운 페이지를 요청할 때마다
서버에서 렌더링된 정적 리소스가 다운로드 됨

페이지를 이동하거나 새로고침 시 전체 페이지를 다시 렌더링



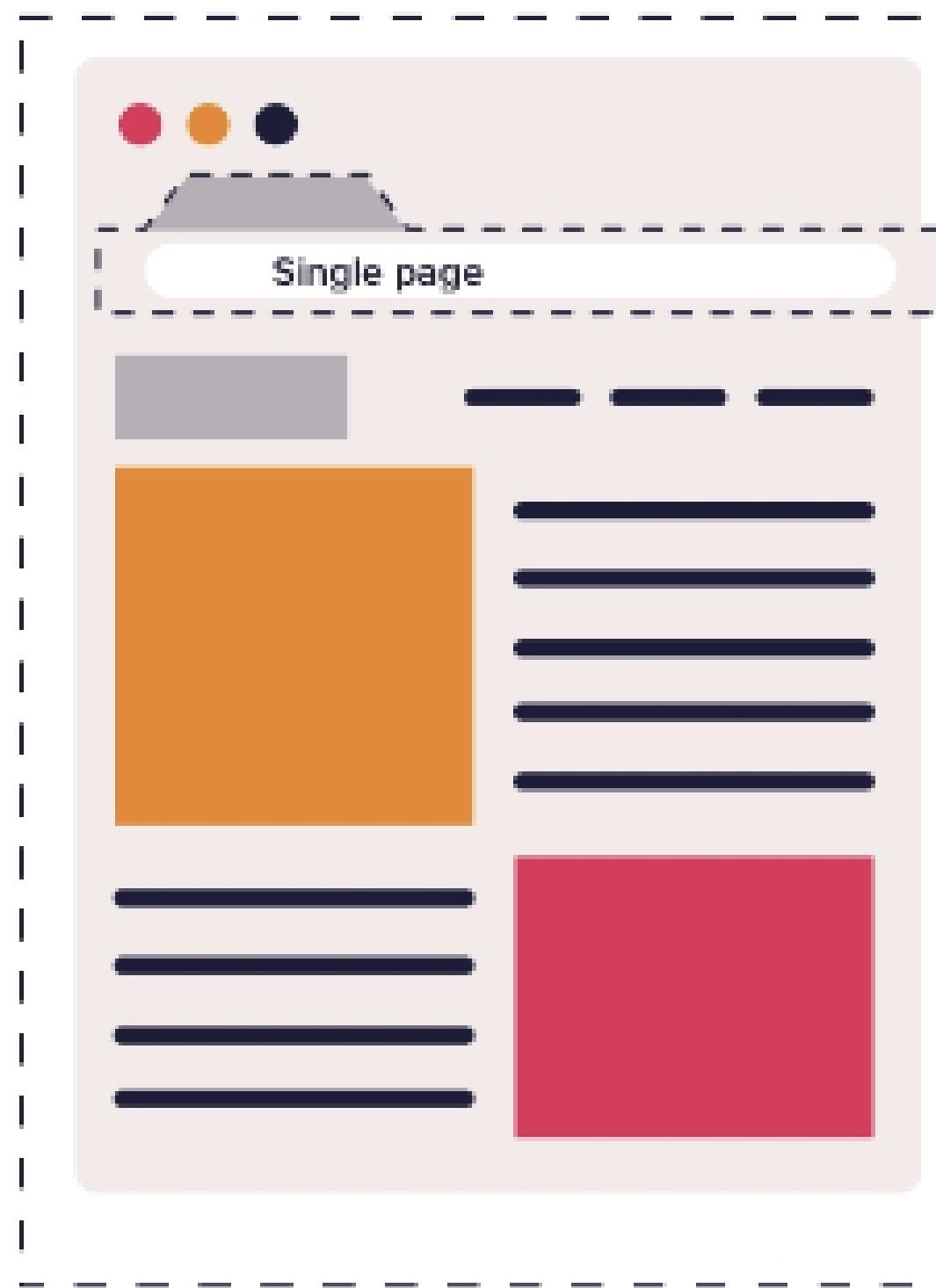
장점

- SEO(Search Engine Optimization)에 유리
- 초기 로딩 시간이 상대적으로 짧다

단점

- 페이지 이동시 깜빡임
- 페이지 요청시 새로고침 발생 + 불필요한 페이지 렌더링
- 서버 성능에 영향을 많이 받음

SPA란?

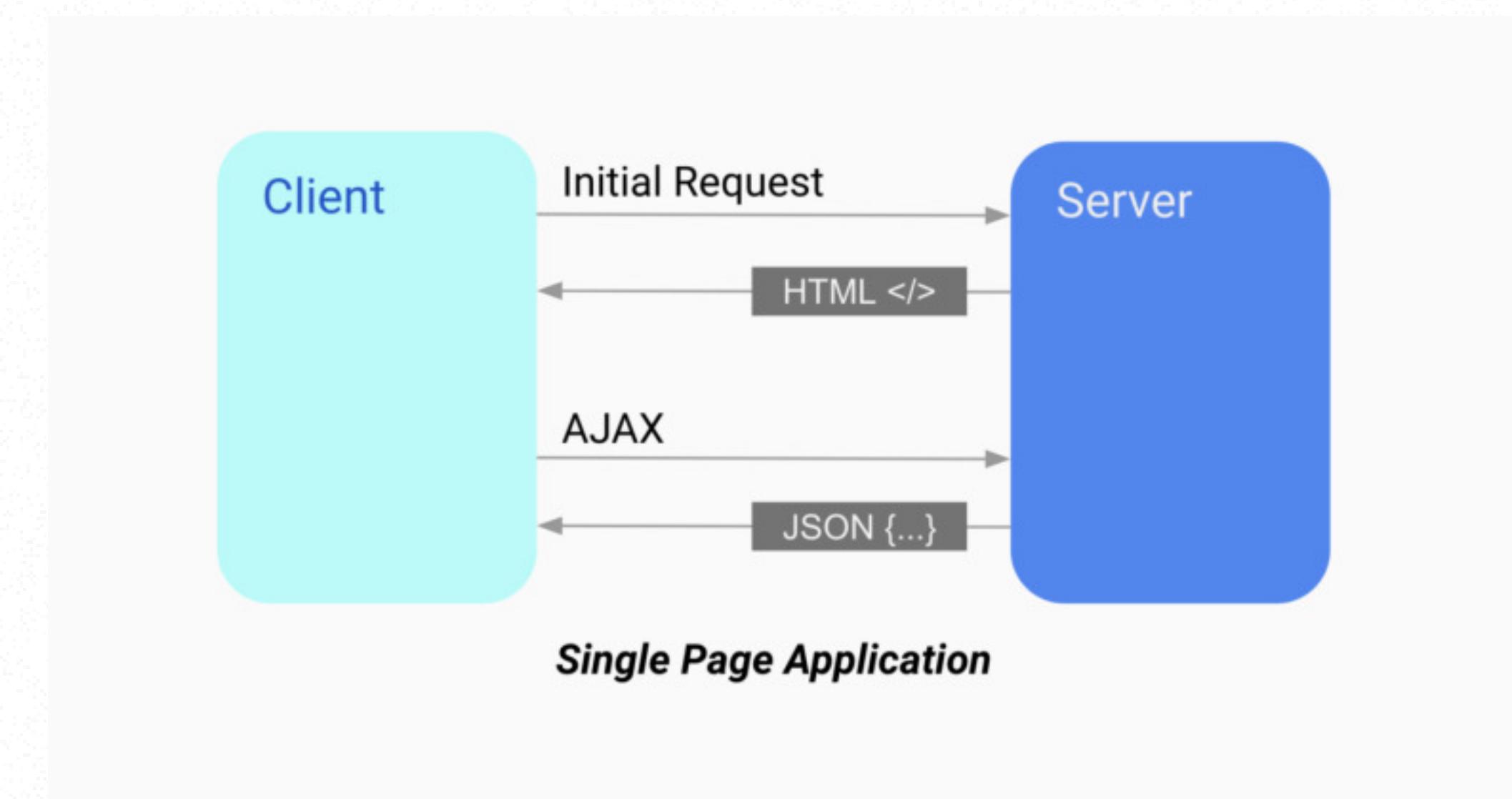


한 개의 페이지로 구성된 애플리케이션

단 한번만 리소스를 로딩
이후에는 데이터를 받아올 때만 서버와 통신

화면의 필요한 부분만 갱신할 수 있음

□ SPA의 장, 단점



장점

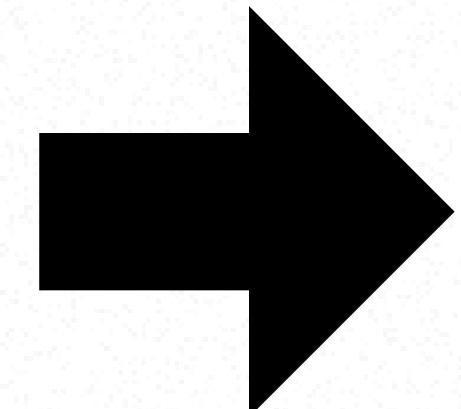
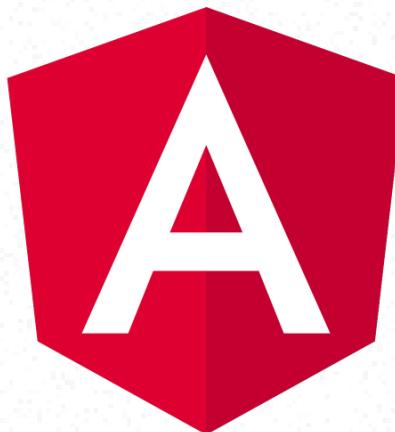
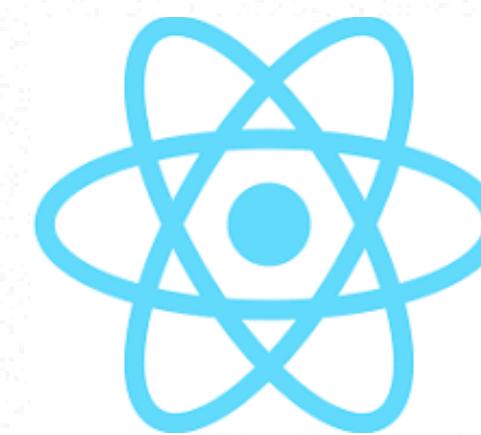
- 자연스러운 사용자 경험(UX)
- 필요한 리소스만 부분적 로딩이 가능 ➡ 성능 ↑
- 컴포넌트 별 개발 용이 ➡ 생산성 ↑

단점

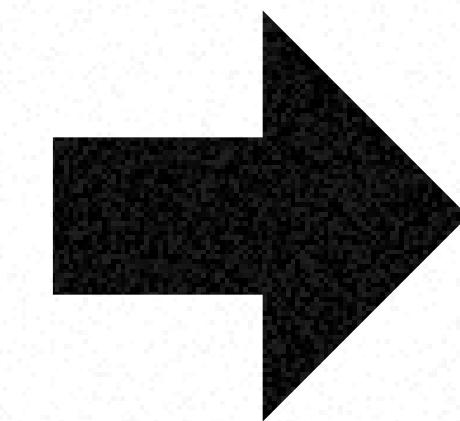
- 초기 로딩 속도가 느림(but, 해결 가능)
- 검색 엔진 최적화가 어려움
- 보안적인 이슈(ex) 사용자 정보 저장)

□ CSR과 SSR, SPA와 MPA



CSR과 SSR, SPA와 MPA

CSR



SSR

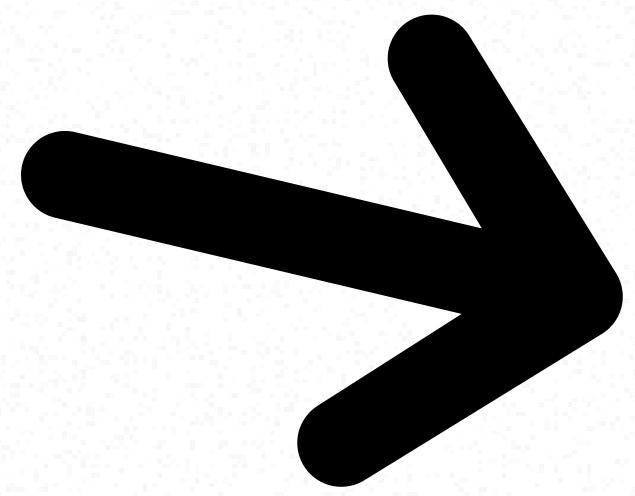
SPA == CSR



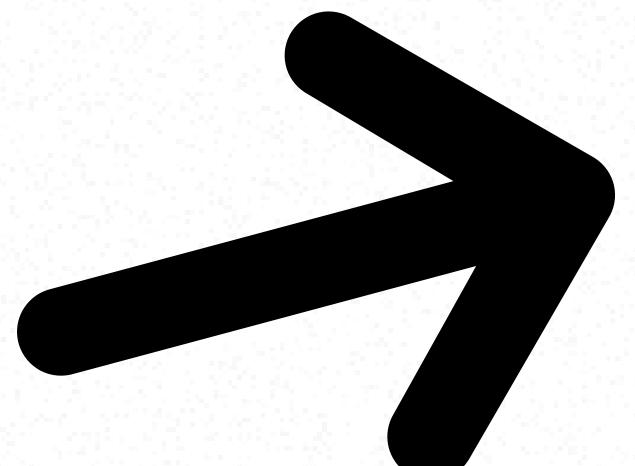
MPA == SSR

CSR이 무조건 좋을까?

CSR

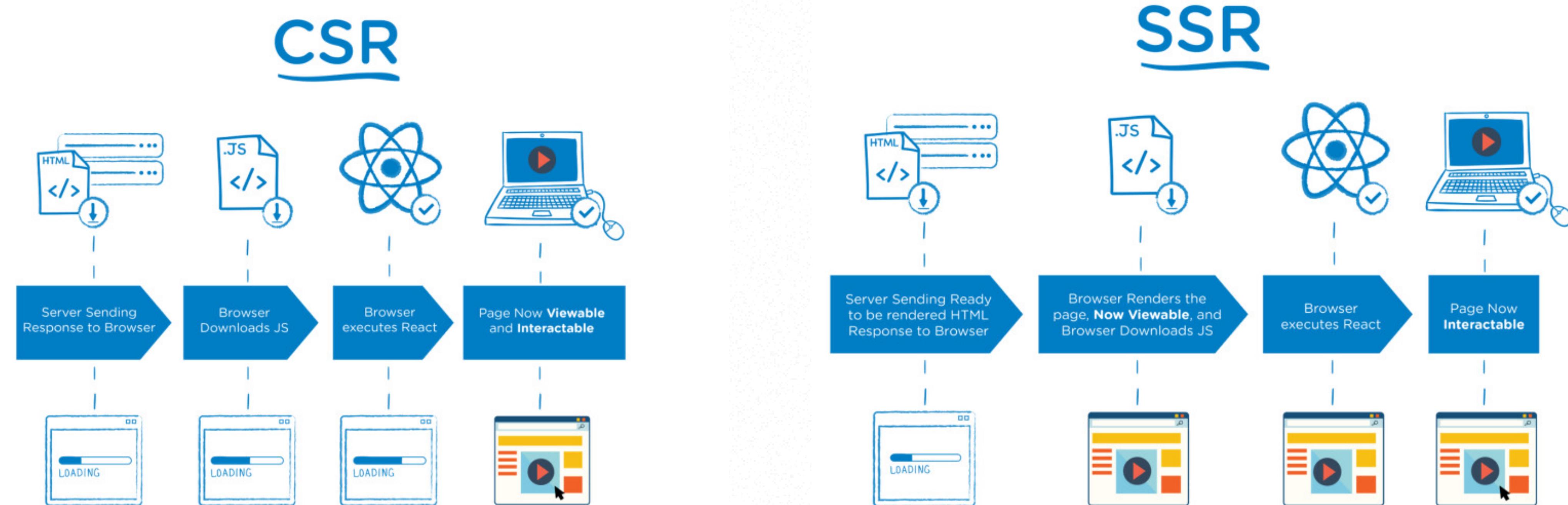


SSR



클라이언트와 서버 중
어느 쪽에서 렌더링 하느냐

□ CSR과 SSR



- 초기 구동 속도가 느리고 SEO에 불리
- but, 이후 과정 처리 속도 빠름 UI / UX 우수

- 초기 구동 속도가 빠르고 SEO에 유리
- but, UI / UX ↓(TTI != TTV)
- 서버 부하 ↑

초기 로딩 속도 보완

- 코드 스플리팅
- tree-shaking chunk 분리

SEO 개선

- pre-rendering

최근엔? CSR 애플리케이션에 SSR 도입 ➡ Next.js

□ CSR의 단점을 보완할 방법



초기렌더링에 SSR을 사용하고 이후 렌더링 과정은 CSR방식을 사용하는 앱 ?

□ 어떻게 하면 적재적소에 쓸 수 있을까?



서비스 성격에 따라 적절하게 사용하자!

검색 엔진에

노출될 필요가 없는 페이지



ex) 관리자 페이지, 개인정보

검색 엔진 최적화,

모든 사람이 같은 화면을 보는 페이지



ex) 회사 홈페이지, 블로그

검색 엔진 최적화, 빠른 인터랙션,

잦은 업데이트를 하는 페이지



ex) 커머스 사이트, 검색 포털