



라우팅

클라이언트 사이드 vs. 서버 사이드 라우팅

서버 사이드 라우팅이란 사용자가 방문하는 URL 경로에 따라 서버가 응답을 보내는 것을 의미합니다. 전통적인 서버 렌더링 웹 앱에서 링크를 클릭하면, 브라우저는 서버로부터 HTML 응답을 받아 전체 페이지를 새로운 HTML로 다시 로드합니다.

하지만 **싱글 페이지 애플리케이션** (SPA)에서는 클라이언트 사이드 자바스크립트가 네비게이션을 가로채고, 동적으로 새로운 데이터를 가져와 전체 페이지를 새로 고침하지 않고 현재 페이지를 업데이트할 수 있습니다. 이는 특히 사용자가 오랜 시간 동안 여러 상호작용을 수행하는 실제 "애플리케이션"과 같은 사용 사례에서 더욱 빠른 사용자 경험을 제공합니다.

이러한 SPA에서는 "라우팅"이 브라우저에서 클라이언트 사이드에서 이루어집니다. 클라이언트 사이드 라우터는 **History API**나 **hashchange** 이벤트와 같은 브라우저 API를 사용하여 애플리케이션의 렌더링된 뷰를 관리하는 역할을 합니다.

공식 라우터

[Vue School](#)에서 무료 비디오 강좌 시청하기

Vue는 SPA를 구축하는 데 매우 적합합니다. 대부분의 SPA에서는 공식적으로 지원되는 **Vue Router** 라이브러리를 사용하는 것이 권장됩니다. 자세한 내용은 Vue Router의 [문서](#)를 참고하세요.

간단한 라우팅 직접 구현하기



적 컴포넌트와 브라우저의 `hashchange` 이벤트 또는 **History API**를 감지하여 현재 컴포넌트 상태를 업데이트하는 방식으로 구현할 수 있습니다.

다음은 최소한의 예시입니다:

```
<script setup>
import { ref, computed } from 'vue'
import Home from './Home.vue'
import About from './About.vue'
import NotFound from './NotFound.vue'

const routes = {
  '/': Home,
  '/about': About
}

const currentPath = ref(window.location.hash)

window.addEventListener('hashchange', () => {
  currentPath.value = window.location.hash
})

const currentView = computed(() => {
  return routes[currentPath.value.slice(1) || '/'] || NotFound
})
</script>

<template>
  <a href="#/">Home</a> |
  <a href="#/about">About</a> |
  <a href="#/non-existent-path">Broken Link</a>
  <component :is="currentView" />
</template>
```

vue

 [Playground에서 실행해보기](#)

 [GitHub에서 이 페이지 편집](#)

[< Previous](#)

틀링

[Next >](#)

상태 관리