Engineer

Next.js

N фреймворк React для створення вебзастосунків

Що це таке

Next.js — це фреймворк, що обгортає React, бібліотеку для побудови інтерфейсів із компонентів, і додає те, що React лишає кожному застосунку: маршрутизацію, рендеринг на сервері, отримання даних і вже налаштовану збірку. Пакет і його команда звуться `next`, як у `next dev` і `next build`.

Уперше його випустили 2016 року, за словами авторів, щоб дати простий спосіб рендерити застосунки React на сервері. Ранній дизайн спирався на кілька принципів: робота без налаштувань, файлова система як API, тож файл у теці `pages/` ставав маршрутом, автоматичний рендеринг на сервері й поділ коду, а отримання даних лишалося на розсуд розробника. 2023 року примітки до випуску дякували понад 2 600 розробникам разом з основною командою у Vercel.

Next.js 13 від 25 жовтня 2022 року додав теку `app` у бета-версії, побудовану на React Server Components, із вкладеними макетами й потоковою передачею. Next.js 13.4 від 4 травня 2023 року оголосив цей App Router стабільним, і відтоді документація описує два маршрутизатори. Pages Router вона називає початковим маршрутизатором, який досі підтримують і вдосконалюють. Вони працюють навіть на різних версіях React: App Router має вбудовані canary-випуски React, а Pages Router використовує ту версію, яку вказано в `package.json` проєкту. В App Router сторінки й макети рендеряться на сервері, якщо файл не обирає браузер директивою `"use client"`.

Next.js 16 вийшов 21 жовтня 2025 року. Він зробив Turbopack типовим бандлером, перейменував файл `middleware.ts` на `proxy.ts` і запровадив Cache Components — модель, у якій директива `"use cache"` позначає те, що можна кешувати, а все інше рендериться під час запиту. Next.js 16.4 від 6

жовтня 2026 року радить Cache Components для кожного застосунку, вмикає їх у нових проєктах, створених через `create-next-app`, і обіцяє, що вони стануть типовими в Next.js 17.

Застосуйте поняття до рішення

Застосуйте «Що таке Next.js?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Інтерфейси, що швидко завантажуються, працюють на телефоні й поводяться однаково в усіх браузерах ваших клієнтів.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Керувала full-stack-розробкою GIS-карт — full-stack-картографія — від бази даних до браузера

Керувала full-stack-розробкою GIS-карт Зв'язатися з нами →

Чому це важливо

Рішення, на яке відповідає Next.js, таке: чи віддати фреймворку маршрутизацію, рендеринг і сервер застосунку на React, чи зібрати ці частини самотужки й лишити вибір за собою.

Фреймворк пропонує домовленості замість конфігурації. Маршрути беруться з файлів, бандлери й компілятори налаштовуються автоматично, а кожен маршрут можна зібрати заздалегідь, рендерити на сервері під час кожного запиту або рендерити в браузері. Server Components тримають доступ до даних і секрети на сервері й надсилають у браузер менше JavaScript. Для застосунку, що поєднує публічні сторінки із зоною після входу та власним бекендом, саме цей вибір для кожного маршруту приваблює найбільше.

Перша ціна — сервер із Node.js 20.9 або новішим для Next.js 16. Параметр `output: 'export'` записує по одному HTML-файлу на маршрут для будь-якого статичного вебсервера, і посібник подає це як спосіб почати статично й додати серверні можливості пізніше. Проте його перелік того, що потребує сервера Node.js, охоплює чимало з того, чим Next.js вирізняється: Proxy, Server Actions, файли cookie, переписування URL, переспрямування, заголовки, інкрементну статичну регенерацію (ISR) і оптимізацію зображень типовим завантажувачем.

Друга ціна — темп. За рік після 16.0 вийшло ще чотири випуски з новими можливостями, і модель кешування от-от знову змінить типову поведінку. Примітки до 16.4 кажуть, що Cache Components замінюють неявне

кешування попередніх версій App Router, тож код, написаний для Next.js 13–15, кешує за іншими правилами. Посібники в мережі теж поділені між двома маршрутизаторами.

Третя ціна — оновлення стають роботою з безпеки. 3 грудня 2025 року проєкт оприлюднив попередження про CVE-2025-66478, критичну вразливість у протоколі React Server Components, і закликав негайно оновитися до виправлених версій, а між 30 вересня і 14 жовтня 2026 року блог опублікував ще три дописи про безпеку. Команда, що обирає Next.js, бере на себе обов'язок уважно стежити за його випусками. Команда, що публікує документи, однакові для кожного відвідувача, платить за серверну модель, якою майже не користується.

Як це виглядає на практиці

У роботі засновниці в NextMariner фронтенд на Next.js 16 обирає режим для кожної частини, і це вибір для кожного маршруту в дії. Маркетингові й публічні сторінки генеруються статично, справді інтерактивні частини й далі рендеряться на клієнті, а в дашборді для автентифікованих користувачів і на великих сторінках-каталогах фабрика `createPrefetchedServerPage` отримує дані сторінки на сервері й передає їх клієнту, який відразу має потрібні дані. Сторінки переносять на фабрику по одній, і сторінка про цю роботу чесно визнає, що решта застосунку для автентифікованих користувачів досі рендериться на клієнті й чекає свого часу: це «перенесення з робочим механізмом і очевидною наступною сторінкою, а не завершений перехід». App Router задумано саме для такого переходу: примітки до 13.4 описують його поступове впровадження.

Багаторівневий набір тестів відводить фронтенду на Next.js окремий рівень: 543 специфікації Vitest і Playwright, зокрема 48 файлів наскрізних тестів і 30 браузерних специфікацій, поряд з окремими рівнями для API на Go і бази даних. Поведінку кожного рівня перевіряють там, де вона живе, а не все ззовні через браузер, і у фреймворку, де одне дерево компонентів працює частково на сервері, а частково в браузері, це важить більше, ніж зазвичай.

Коли засновниця, працюючи в NextMariner, посилила захист застосунку, захисні механізми, звернені до браузера, опинилися в `proxy.ts`, перейменованому middleware. Proxy виконується раніше за маршрут, тож саме тут кожен запит отримує власний nonce, який іде в Content-Security-Policy поряд зі `strict-dynamic`; тут же виставляються HSTS і SameSite cookies.

Права доступу живуть деінде: тариф, роль, організацію й прив'язку до судна сервер повторно перевіряє під час кожного запиту, а обмеження на фронтенді вважаються суто питанням UX. Такий поділ узгоджується з довідником з Proxy, який називає цю можливість крайнім засобом і радить перевіряти авторизацію всередині кожної Server Function, а не покладатися лише на Proxy.

Де виникають проблеми

Після додавання переспрямування на вхід сторінка входу завантажується без стилів. Без `matcher` Proxy працює на кожному запиті, зокрема для зібраних файлів у `_next/static`, оптимізованих зображень у `_next/image` і всього в `public/`, тож переспрямування, призначене для сторінок, перехоплює й CSS, JavaScript і зображення, потрібні сторінці входу. Дайте Proxy matcher, що пропускає ці шляхи:

```
// proxy.ts
export const config = {
    matcher: ["/((?!_next/static|_next/image|favicon.ico).*)"],
};
```

Файли з `public/` не мають спільного префікса, тож додайте і їхні шляхи.

Дія форми спрацьовує для користувача, який не ввійшов, на маршруті, що виглядає захищеним. Server Functions не мають власних маршрутів: вони надходять як POST-запити на сторінку, де їх використано, тож matcher, який виключає шлях із перевірки Proxy, виключає з неї й функції на ньому. Довідник з Proxy попереджає, що зміна matcher або рефакторинг, який переносить функцію на інший маршрут, може непомітно зняти покриття Proxy. Перевіряйте сесію й права користувача всередині кожної Server Function і вважайте Proxy зовнішнім шаром, а не єдиним.

Користувач зберігає зміну, а сторінка досі показує старе значення. У Next.js 16 `revalidateTag(tag, 'max')` дає позначеним записам поведінку stale-while-revalidate: кешована копія й далі віддається, поки отримується свіжа, що пасує вмісту, для якого припустима узгодженість із затримкою, і дивує людину, яка щойно натиснула «Зберегти». Викликайте `updateTag(tag)` з тієї

Server Action, що внесла зміну: вона негайно анулює кешований запис і читає свіжі дані в межах того самого запиту. Стара форма з одним аргументом, `revalidateTag(tag)`, оголошена застарілою й тепер дає помилку TypeScript.

Codemod для оновлення завершується, а збірка досі падає на `params` або `cookies()`. Next.js 15 зробив API запиту асинхронними з тимчасовою синхронною сумісністю, а Next.js 16 її прибирає. Codemod `upgrade` виконує не всі потрібні перетворення коду, тож запустіть також `npx @next/codemod@canary next-async-request-api .`, а потім `npx next typegen`, щоб згенерувати допоміжні типи `PageProps`, `LayoutProps` і `RouteContext`. Те саме оновлення прибирає `next lint`, а `next build` більше не запускає лінтер, тож додайте ESLint або Biome окремим кроком.

Що почитати далі

- Next.js Docs — офіційна документація з перемикачем між версіями кожної сторінки для App Router і Pages Router.
- Next.js 16 — випуск, що приніс Cache Components, `proxy.ts` і Turbopack як типовий бандлер, разом з несумісними змінами.
- How to upgrade to version 16 — codemod-и, вимоги до версій і кожен вилучений чи перейменований API.
- proxy.js — matcher-и, місце Proxy в порядку обробки запиту і причина, з якої middleware перейменували.
- Next.js 16.4 — аргументи на користь Cache Components і план зробити їх типовими в Next.js 17.
- How to create a static export — що записує `output: 'export'` і яким можливостям потрібен сервер.

Словник

Обирайте за фактичними умовами

Опишіть робочий процес, наявні інтерфейси та користувачів, яким потрібна зміна. Ми можемо визначити перевірюваний результат і спосіб передачі, який дасть вашій команді змогу підтримувати його.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке Next.js?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/nextjs/