Engineer

Next.js

N et React-rammeverk for å bygge webapper

Hva det er

Next.js er et rammeverk rundt React, biblioteket for å bygge brukergrensesnitt av komponenter, og det leverer de delene React overlater til hver enkelt applikasjon: ruting, rendering på serveren, henting av data og et bygg som er ferdig konfigurert. Både pakken og kommandoen heter `next`, som i `next dev` og `next build`.

Første versjon kom i 2016, og ifølge forfatterne var målet å gjøre det enkelt å rendre React-applikasjoner på serveren. Den tidlige utformingen hvilte på noen få prinsipper: null oppsett, filsystemet som API, slik at en fil i mappen `pages/` ble en rute, automatisk rendering på serveren og oppdeling av koden, og henting av data overlatt til utvikleren. I 2023 takket utgivelsesnotatene over 2 600 utviklere i tillegg til et kjerneteam hos Vercel.

Next.js 13 kom 25. oktober 2022 og la til en `app`-mappe i beta, bygget på React Server Components, med nøstede layouter og strømming. Next.js 13.4 erklærte 4. mai 2023 at denne App Router var stabil, og siden da har dokumentasjonen beskrevet to rutere. Pages Router omtales der som den opprinnelige ruteren, som fortsatt støttes og fortsatt blir bedre. De to kjører til og med hver sin React: App Router leveres med canary-utgivelser av React bygget inn, mens Pages Router bruker den versjonen som står i prosjektets `package.json`. I App Router rendres sider og layouter på serveren, med mindre en fil melder seg inn i nettleseren med direktivet `"use client"`.

Next.js 16 kom 21. oktober 2025. Den gjorde Turbopack til standardverktøyet for bunting, ga filen `middleware.ts` det nye navnet `proxy.ts` og innførte Cache Components, en modell der direktivet `"use cache"` markerer hva som kan mellomlagres, mens alt annet rendres når forespørselen kommer. Next.js 16.4,

som kom 6. oktober 2026, anbefaler Cache Components for alle apper, slår dem på for nye prosjekter laget med `create-next-app` og sier at de blir standard i Next.js 17.

Bruk begrepet i en beslutning

Bruk «Hva er Next.js?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Grensesnitt som laster raskt, fungerer på mobilen og oppfører seg likt i alle nettlesere kundene dine måtte bruke.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Styrte fullstack-utviklingen av GIS-kart — fullstack-kartutvikling, fra database til nettleser

Styrte fullstack-utviklingen av GIS-kart Ta kontakt →

Hvorfor det er viktig

Valget Next.js svarer på, er om et rammeverk skal eie rutingen, renderingen og serveren i en React-applikasjon, eller om dere setter sammen delene selv og beholder valgene.

Det Next.js tilbyr, er konvensjoner i stedet for konfigurasjon. Rutene kommer fra filer, bundlere og kompilatorer settes opp automatisk, og hver rute kan bygges på forhånd, rendres på serveren ved hver forespørsel eller rendres i nettleseren. Server Components holder datatilgang og hemmeligheter på serveren og sender mindre JavaScript til nettleseren. For en applikasjon som blander offentlige sider med et innlogget område og en egen backend, er nettopp dette valget per rute hovedgrunnen til å velge rammeverket.

Den første kostnaden er en server, som for Next.js 16 må kjøre Node.js 20.9 eller nyere. Med `output: 'export'` skrives én HTML-fil per rute, klar for en hvilken som helst statisk webserver, og veiledningen presenterer det som en måte å starte statisk på og legge til serverfunksjoner senere. Listen i samme veiledning over hva som krever en Node.js-server, dekker likevel mye av det som skiller Next.js fra andre: Proxy, Server Actions, informasjonskapsler, omskrivinger, omdirigeringer, headere, Incremental Static Regeneration og bildeoptimalisering med standardlasteren.

Den andre kostnaden er tempoet. Fire funksjonsutgivelser fulgte etter 16.0 i løpet av ett år, og modellen for caching er i ferd med å bytte standard igjen. Ifølge notatene til 16.4 erstatter Cache Components den implisitte cachingen i

tidligere versjoner av App Router, så kode skrevet for Next.js 13–15 mellomlagrer etter andre regler. Veiledninger du finner på nettet, er dessuten delt mellom de to ruterne.

Den tredje kostnaden er at oppgraderinger er sikkerhetsarbeid. Den 3. desember 2025 publiserte prosjektet et sikkerhetsvarsel om CVE-2025-66478, en kritisk sårbarhet i protokollen til React Server Components, og ba alle oppgradere til rettede versjoner med en gang, og mellom 30. september og 14. oktober 2026 hadde bloggen tre nye innlegg om sikkerhet. Et team som tar i bruk Next.js, forplikter seg til å følge utgivelsene tett. Et team som publiserer dokumenter som ser like ut for alle besøkende, betaler for en servermodell det knapt bruker.

Slik ser det ut i praksis

Next.js 16-frontenden i grunnleggerens arbeid hos NextMariner rendrer hver flate på sin måte, og det er valget per rute satt i arbeid. Markedsføringssider og offentlige sider genereres statisk, de virkelig interaktive delene forblir klientrendret, og på det innloggede dashbordet og de store katalogoversiktene henter en fabrikk, `createPrefetchedServerPage` , sidens data på serveren og gir dem til klienten, som dermed allerede har dataene sine. Sidene flyttes over på fabrikken én om gangen, og beskrivelsen er åpen om at resten av den innloggede appen fortsatt er klientrendret og venter på tur etter den: «en migrering med en fungerende mekanisme og en åpenbar neste side, og ikke en fullført opprydding». App Router ble laget for akkurat den typen overgang, og utgivelsesnotatene for 13.4 beskriver hvordan den kan tas i bruk gradvis.

En lagdelt testsuite gir en Next.js-frontend et lag for seg selv: 543 Vitest- og Playwright-spesifikasjoner, blant dem 48 ende-til-ende-filer og 30 nettleserspesifikasjoner, ved siden av egne lag for Go-API-et og databasen. Hvert lag kontrolleres der oppførselen hører hjemme, i stedet for at alt testes gjennom en nettleser, og det betyr noe i et rammeverk der komponenttreet kjører dels på serveren og dels i nettleseren.

Da grunnleggeren i sitt arbeid hos NextMariner herdet en applikasjon, havnet beskyttelsene som vender mot nettleseren, i `proxy.ts` , mellomvaren med nytt navn. Der får hvert svar HSTS, SameSite-cookies og en Content-Security-Policy med `strict-dynamic` og en nonce som er ny for hver forespørsel. Autorisasjonen ligger et annet sted: abonnementsnivå, rolle, organisasjon og avgrensning per skip går serveren gjennom igjen for hver forespørsel, og det en Next.js-side

skjuler eller sperrer, regnes bare som UX. Den delingen stemmer med referansen for Proxy, som kaller funksjonen en siste utvei og ber utviklere kontrollere autorisasjon inne i hver Server Function i stedet for å stole på Proxy alene.

Hvor det går galt

Etter at en omdirigering til innlogging er lagt til, lastes innloggingssiden uten stiler. Proxy kjører på hver eneste forespørsel så lenge den mangler en `matcher`, også på bygde filer under `_next/static`, optimaliserte bilder under `_next/image` og alt i `public/`, så en omdirigering som var ment for sider, fanger også CSS-en, JavaScript-koden og bildene innloggingssiden trenger. Gi Proxy en matcher som hopper over disse stiene:

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

Filer fra `public/` deler ikke et felles prefiks, så legg til stiene deres også.

En skjemahandling virker for en utlogget bruker på en rute som ser beskyttet ut. Server Functions har ikke egne ruter: de kommer som POST-forespørsler til siden der de brukes, så en matcher som hopper over en sti, hopper også over funksjonene der. Referansen for Proxy advarer om at en endret matcher, eller en omskriving som flytter en funksjon til en annen rute, kan fjerne dekningen fra Proxy uten at det merkes. Sjekk økten og brukerens rettigheter inne i hver Server Function, og behandle Proxy som et ytre lag, aldri som det eneste.

En bruker lagrer en endring, og siden viser fortsatt den gamle verdien. I Next.js 16 gir `revalidateTag(tag, 'max')` merkede oppføringer oppførselen stale-while-revalidate: den mellomlagrede kopien serveres videre mens en fersk hentes, og det fungerer godt for innhold som tåler å bli oppdatert litt etter hvert, men overrasker den som akkurat trykket Lagre. Kall `updateTag(tag)` fra den Server Action som gjorde endringen; den lar oppføringen utløpe og leser ferske data i samme forespørsel. Den gamle `revalidateTag(tag)` med ett argument er avviklet og gir nå en feil i TypeScript.

Oppgraderingens codemod blir ferdig, og bygget feiler fortsatt på `params` eller `cookies()`. Next.js 15 gjorde forespørsels-API-ene asynkrone, med midlertidig synkron kompatibilitet, og Next.js 16 fjerner den. Codemoden `upgrade` dekker bare en del av omleggingene, så kjør også `npx @next/codemod@canary next-async-request-api .` og deretter `npx next typegen` for å generere hjelpetypene `PageProps`, `LayoutProps` og `RouteContext`. Den samme oppgraderingen fjerner `next lint` og får `next build` til å slutte å kjøre linting, så legg til ESLint eller Biome som et eget steg.

Les mer

- Next.js Docs — den offisielle dokumentasjonen, med en bryter mellom versjonene for App Router og Pages Router på hver side.
- Next.js 16 — utgivelsen som innførte Cache Components, `proxy.ts` og Turbopack som standard for bunting, med endringene som bryter med eldre kode.
- How to upgrade to version 16 — codemodene, versjonskravene og hvert API som er fjernet eller har fått nytt navn.
- proxy.js — matchere, hvor Proxy står i rekkefølgen for en forespørsel, og hvorfor mellomvaren fikk nytt navn.
- Next.js 16.4 — argumentene for Cache Components og planen om å gjøre dem til standard i Next.js 17.
- How to create a static export — hvilke funksjoner som krever en server, og hva `output: 'export'` skriver.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er Next.js?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/nextjs/