Engineer

App

et program, folk bruger til at få en opgave løst

Hvad det er

En app er et program, som et menneske åbner og arbejder i: det har et ikon at starte fra, sit eget vindue eller skærmbillede og en opgave, brugeren kan genkende, som at sende en besked, slå en køreplan op eller indberette en udgift. Ordet er en forkortelse af det engelske application, og på dansk lever både app, i bestemt form appen, og applikation side om side. Den lange form ses mest i formelle tekster og i sammensætninger som webapplikation, og udviklere siger ofte klient, når appen er den synlige ende af en tjeneste, der kører et andet sted.

En app er applikationssoftware, modsat den systemsoftware, der holder selve maskinen kørende. NIST's ordliste samler flere definitioner og bemærker, at application og software application ofte bruges som synonymer, og én af dem regner også løn- og regnskabssystemer for applikationer. Et bibliotek står på den anden side af skellet: kode, som andre programmer kalder, og som intet menneske nogensinde åbner.

Apps falder i tre store familier:

- **Native apps** skrives til én platform i platformens eget sprog og værktøj, for eksempel Swift og SwiftUI på en Mac, og installeres fra en butik eller som en signeret download.
- **Webapps** kører i en browser og nås via et link. Et web application manifest, en lille JSON-fil, der fortæller browseren appens navn, ikoner, start-URL og foretrukne visning, gør det muligt at lægge en webapp på hjemmeskærmen eller i docken, så den åbner i sit eget vindue. En service worker, et script som browseren kører mellem siden og netværket, er det, der lader den fortsætte uden net. Tilsammen kaldes de to som regel en progressiv webapp, forkortet PWA.
- **Tværplatformsskaller** pakker én kodebase ind til flere styresystemer, ofte ved at lægge en browsermotor ind i selve appen.

Brug begrebet i en beslutning

Brug »Hvad er en app?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Et site, der er hurtigt, kan findes og kan redigeres af jeres egne folk, uden en regning for hver ændret sætning.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har udviklet 100 webapplikationer med HTML/HTML5 — 100 webapplikationer bygget

Har udviklet 100 webapplikationer med HTML/HTML5 Kontakt os →

Hvorfor det betyder noget

Bag de fleste projekter, der skal ende som en app, ligger spørgsmålet om, hvilken familie man skal bygge i. Svaret afhænger mest af brugere og distribution og kun lidt af koden.

En webapp kræver én kodebase og når alle med en browser gennem et link. En opdatering er ude, så snart serveren er: næste genindlæsning henter den, og der står ingen butiksgennemgang mellem en rettelse og de mennesker, der venter på den. Prisen er dybde. En browser giver adgang til mindre af enheden, end styresystemet gør, og hver browser præsenterer installationen på sin egen måde. På iOS dukker installation aldrig op som et forslag, så et hold, der vil have sin webapp ud på hjemmeskærmene, må selv fortælle folk hvordan.

En native app får platformens egne kontroller, tastaturadfærd, tilgængelighedsstøtte og ydeevne og adgang til alt, hvad systemet tilbyder. Den koster en kodebase pr. platform, folk, der kender hver af dem, og en distributionsproces: gennemgang i en butik eller, uden for butikken, kodesignering og på macOS notarisering, som Apple siden macOS 10.15 kræver af software, der distribueres med et Developer ID. Nogen skal også drive opdateringskanalen, så længe appen findes.

En tværplatformsskal ligger imellem de to: én kodebase og en app, der kan installeres, betalt med en medbragt runtime og en brugerflade, der ligner platformen uden helt at opføre sig som den.

Det, der som regel afgør sagen, er, hvor brugerne befinder sig, når de åbner appen. Den, der sidder ved et skrivebord med kabelforbindelse, tilgiver meget. Den, der står med en telefon og svag dækning, tilgiver næsten intet, og for den

bruger er det et krav og ikke pynt, at appen virker uden net. Det andet afgørende spørgsmål er, hvem der udgiver opdateringer i det tredje år, for den app, der er billigst at bygge, kan blive den dyreste at holde i live.

Sådan ser det ud i praksis

På én platform er mange af brugerne søfolk og feltpersonale, der arbejder fra telefoner og ofte er helt uden dækning. Svaret blev en webapp, der kan installeres: fuld understøttelse af progressive webapps giver den ikoner til hjemmeskærmen i de størrelser, platformene beder om, en startskærm og temafarver. Dens service worker sættes op via pluginet next-pwa, Workboxs runtime-caching leverer skrifttyper, billeder, lyd, video og filer fra CDN'er cache-first sammen med HTTP-headere til cachestyring, og installation og brug uden net blev afprøvet over HTTPS på rigtige enheder. Folk kan installere den, blive ved med at bruge den, når nettet forsvinder, og få gentagne besøg leveret fra cachen.

En beskedprotokol, hvis desktopklient på macOS var en tværplatformsskal, gik den modsatte vej med en native macOS-klient i Swift 6 og SwiftUI på 11.141 linjer fordelt på 53 filer, hvor protokollens Rust-kerne er linket ind som et statisk arkiv bygget fra en fastlåst revision. Klienten starter som et Mac-program, bruger platformens egne kontroller og har ingen indbygget browser med. Siden siger også, hvor arbejdet holder op: det er ikke en udgivet version, der er hverken et notariseret build eller en opdateringskanal, og brugerfladen findes på ét sprog.

Dette site kan også installeres som app, og installationssiden gennemgår det browser for browser: ingen butik, ingen konto, ingen særskilt app-pakke at hente, bare et ikon og et vindue for sig selv. Safari på en iPhone har det under Del. Firefox på Windows kan fra version 143 føje websites til, så de åbnes i deres egne webappvinduer. Hvis Firefox er installeret fra Microsoft Store, er funktionen tilgængelig fra version 150. Mozilla tilbyder ikke denne funktion på macOS eller Linux. Siden nævner også grænsen: brug uden net kræver en service worker, sitet leverer ingen JavaScript til afvikling i browseren og registrerer ingen service worker, og derfor skal den installerede app have forbindelse ligesom en almindelig fane.

Hvor det går galt

Skærmen bliver hvid, når dækningen forsvinder

Brugere i felten melder om et hvidt vindue eller browserens egen offlineside. Appen blev designet og testet på kontorets hurtige forbindelse, og hver skærm henter det, den skal bruge, i samme øjeblik den åbnes. Beslut, hvilke skærme der skal virke uden net, læg dem og deres filer i cache med en service worker, og test på rigtige enheder med netværket slået fra i stedet for i en desktopbrowsers simulering.

Brugerne installerer en webapp og forventer en offlinekopi

Nogen lægger sitet på hjemmeskærmen, åbner det i et tog og møder den samme fejl, som en browserfane ville vise. Et manifest gør et site installerbart, mens brug uden net kun kommer fra en service worker, og det er to forskellige beslutninger. Lever enten workeren, eller skriv tydeligt i installationsvejledningen, at den installerede app kræver forbindelse.

En ny version går ud, og brugerne beholder den gamle

Efter en udrulning kører nogle folk sidste uges kode i dagevis. Service workeren leverer appens sider cache-first uden noget, der markerer en ny version, eller en ny worker venter, indtil alle faner, som den gamle styrer, er lukket. Behold cache-first til filer, der er ens fra forespørgsel til forespørgsel, som skrifttyper, medier og filer med versionsnummer i navnet, hent dokumenter fra netværket først, og vis en opfordring til at genindlæse, når en ny version er klar.

Skallen føles fremmed på skrivebordet

Mac-brugere klager over menuer, genveje og markering af tekst, og appen bruger hukommelse som en browser, fordi den indeholder en. En medbragt runtime efterligner platformen i stedet for at bruge den. Hvor oplevelsen på skrivebordet er selve produktet, så sammenlign en native klient med skallen, og hold den fælles logik i et kernebibliotek, som alle klienter kalder gennem en stabil grænseflade.

Distributionen bliver gemt til sidst

Appen virker på hver udviklers maskine, og den første kunde på macOS møder en advarsel fra Gatekeeper i stedet for appen. En rettelse bliver udgivet, og halvdelen af brugerne bliver ved med at køre den fejlbehæftede version.

Signering, notarisering og opdateringer blev behandlet som pakkearbejde til den sidste uge. Byg signering og notarisering ind i byggeprocessen fra første udgivelse, og afgør, hvordan opdateringer når ud, før nogen installerer version 1.

Læs mere

- NIST CSRC Glossary: application — de definitioner, NIST's publikationer bruger, fra ethvert system bygget til at opfylde en række krav til løn- og regnskabssystemer.
- W3C Web Application Manifest — JSON-filen, der navngiver en webapp, angiver dens ikoner og start-URL og bestemmer, hvordan den vises, når den åbnes.
- W3C Service Workers — scriptet mellem en webapp og netværket med de cache- og fetch-hændelser, der gør brug uden net muligt.
- Apple Developer: Notarizing macOS software before distribution — hvad notartjenesten tjekker, og hvornår software uden for Mac App Store skal igennem den.



Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er en app?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

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

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/app/