Engineer

macOS

Apples styresystem til computere

Hvad det er

macOS er det styresystem, Apple leverer på hver Mac: skrivebordet og Finder, de medfølgende apps og under dem et Unix-system med terminal, shell og de velkendte kommandolinjeværktøjer. Apple bygger det til sin egen hardware og sælger de to sammen, så i praksis er et valg af macOS et valg af en Mac.

Fundamentet hedder Darwin, og Apple offentliggør kildekoden til det. Systemkernen, XNU, er en hybrid, der forener Mach-kernen fra Carnegie Mellon University med komponenter fra FreeBSD og en driverramme i C++ ved navn IOKit; navnet står for »X is Not Unix«. Meget af brugerlandet kommer også fra FreeBSD, og manualsiderne for `sed` og `date` på en Mac beskriver FreeBSD's udgaver af de to værktøjer. Siden macOS 10.15 Catalina har standardshellen været zsh.

Hardwaren under systemet er skiftet for nylig. I slutningen af 2020 begyndte Apple at flytte Mac'en fra Intel-processorer til sine egne chips, Apple silicon, og hver Mac-serie fra MacBook Air til Mac Pro findes nu med Apple silicon. De chips er arm64, og det får betydning, så snart software bygget på en Mac skal køre på en x86-64-server.

På bærbare og stationære computere holdes macOS op imod Windows og Linux.

Brug begrebet i en beslutning

Brug »Hvad er macOS?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Den bærbare, printeren, login og postkassen klaret hurtigt og i et sprog, man forstår, uden at nogen føler sig dum.

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 etableret en teknisk supportafdeling — 10,000 kunder betjent af en supportafdeling bygget fra bunden

Har etableret en teknisk supportafdeling Kontakt os →

Hvorfor det betyder noget

For en virksomhed, der skriver software, spiller macOS to roller, som er lette at blande sammen: det er den maskine, mange udviklere arbejder på, og det er en platform, kunderne kører software på.

Som arbejdsstation adskiller en Mac sig fra de Linux-servere, koden udrulles til, på tre punkter. Kommandolinjeværktøjerne stammer fra FreeBSD og ikke fra GNU, så samme flag kan betyde to forskellige ting i et script. Processoren er arm64, hvor serveren måske er x86-64. Og der er ingen Linux-kerne på en Mac, mens Linux-containere deler værtens kerne, så værktøjer, der kører dem på en Mac, starter først en virtuel Linux-maskine; Apples eget værktøj `container`, som kræver Apple silicon og macOS 26, giver hver container sin egen lette virtuelle maskine. Intet af det udelukker en Mac. Det betyder, at man tester på det system, man udruller til, frem for at stole på en grøn kørsel på en bærbar.

Som platform rejser macOS spørgsmålet om, hvordan man bygger til den. En skal på tværs af platforme, én kodebase med en indlejret browsermotor, der leveres til Windows, Linux og Mac, holder flere systemer i takt for få penge, men på Mac'en slæber den en browserkøretid med, som et native program ikke har brug for, og tegner sine egne kontroller i stedet for systemets. Et native program i Swift og SwiftUI bruger systemets kontroller og materialer og følger brugerens indstillinger for udseende, accentfarve, kontrast, tekststørrelse og reduceret bevægelse uden ekstra kode. Prisen er en separat klient og folk, der kender platformen. Native betaler sig, når Mac-brugerne er kunderne.

Levering til Mac går også gennem en kontrol. Software, der distribueres uden for Mac App Store, signeres med et Developer ID-certifikat og sendes til Apples notariseringstjeneste, en automatisk scanning for skadeligt indhold og fejl i kodesigneringen, som er noget andet end App Review. Siden macOS 10.15 skal al software, der er bygget efter 1. juni 2019 og distribueres med Developer ID, være notariseret. Apple sender en billet tilbage, som hæftes på softwaren, og offentliggør den samme billet online, hvor Gatekeeper slår den op, første gang brugeren åbner appen. Certifikatet følger med Apple Developer Program, der koster 99 dollar om året. Planlæg medlemskabet og signeringen ind i den første udgivelse.

Sådan ser det ud i praksis

Skrivebordsoplevelsen af et beskedprodukt på macOS var en skal på tværs af platforme, der hverken lignede eller opførte sig som et Mac-program. Produktets velafprøvede Rust-kerne har en C-grænseflade, og Swift kan kalde C, så en native klient i Swift og SwiftUI blev bygget direkte på den grænseflade. Rust-kernen er linket ind som et statisk arkiv fra en fastlåst opstrømsrevision, hvilket gør bygget gentageligt, og en tynd indpakning ejer hver pointer, der krydser mellem sprogene. Klienten starter som et Mac-program, bruger platformens egne materialer og kontroller og har ingen indlejret browser. Siden er ærlig om afstanden: der er ingen notariseret distribution og ingen opdateringskanal, så det er et fungerende program og ikke et produkt, som andre kører. Notariseringen ovenfor er netop det, der står mellem de to.

Platformens udseende kan håndhæves i stedet for at efterlignes. Et designsystem på 48 navngivne tokens i seks grupper bygger farve og materiale på platformens semantiske farver og dens glasmateriale frem for faste værdier, så grænsefladen følger systemets udseende, accentfarven og kontrastindstillingerne, uden at nogen kode holder øje med dem, og tekststilene skalerer med brugerens foretrukne størrelse. En linter afviser et råt tal, hvor et token hører hjemme, en hårdkodet skriftstørrelse og en animation, der ignorerer præferencen for reduceret bevægelse. Siden nævner også sin grænse: tokens gør en grænseflade ensartet, og det er noget andet end at gøre den god.

Komprimeret hukommelse og swap lader en macOS-app vokse langt ud over maskinens fysiske hukommelse. En hændelsesbeskrivelse dokumenterer diagnoserapporter med 111 GB komprimerede sider på en maskine med 36 GB den 2026-09-01 og en separat måling af hukommelsesforbruget, som viste en vækst på omkring 41 MB i sekundet under en løkke med gentagne forbindelsesforsøg. Hændelsesvejen beholdt core-loghændelser i en ubegrænset hovedstrøm og kopierede dem til abonnenter, der ikke brugte de hændelser. Bufferne blev afgrænset, abonnementerne filtreret efter hændelsestype, og logningen fik et hastighedsbudget. En sammenligning af fulde testkørsler registrerede et fald fra 610.996 core-loglinjer til 1.411; det er målinger fra testkørsler, ikke en måling af hukommelsesforbruget efter rettelsen eller et benchmark fra produktion. Afvejningen er udtrykkelig: gentagne diagnostiske hændelser kan kasseres, og den berørte tilstand kan genindlæses i stedet for at beholde ubegrænset mange hændelser. Separate timeouts ved nedlukning krævede en separat rettelse.

Hvor det går galt

Aktivitetsovervågning viser næsten ingen ledig hukommelse på en Mac, der kører helt fint. macOS bruger ellers ubrugt hukommelse som cache for nyligt brugte filer og komprimerer, efterhånden som hukommelsen fyldes, inaktive apps for at give plads til aktive. Brugt hukommelse tæt på den fysiske og et stort tal for komprimeret hukommelse er derfor normalt. Apple skriver selv, at ledig hukommelse ikke nødvendigvis forbedrer computerens ydeevne. Læs i stedet grafen Belastning af hukommelse: grøn betyder, at computeren bruger al RAM effektivt, gul, at den kan få brug for mere på et tidspunkt, og rød, at den har brug for mere. Swap er det tydeligste tegn:

```
sysctl vm.swapusage
vm_stat | grep -E '^Swap(ins|outs)'
```

Tællerne i `vm_stat` løber fra opstart, så sammenlign to aflæsninger med et minuts mellemrum; swapouts, der bliver ved med at stige, betyder, at arbejdet er vokset fra RAM'en.

Et script, der virker på Mac'en, stopper med en mærkelig fejl på Linux-serveren, eller omvendt. Den FreeBSD-afledte `sed` på en Mac læser ordet efter `-i` som et suffiks til sikkerhedskopien, så GNU-vanen `sed -i 's/a/b/' fil` tager udtrykket som suffiks og filnavnet som script. Mac-vanen `sed -i '' …` får GNU `sed` til at læse udtrykket som et filnavn. Et suffiks skrevet sammen med flaget virker begge steder:

```
sed -i.bak 's/^port=80$/port=8080/' app.conf && rm app.conf.bak
```

Den varige løsning er at køre scripts i CI på det system, de møder i produktion.

En app, der deles uden for holdet, bliver blokeret af Gatekeeper, når modtageren åbner den. Appen er signeret til udvikling, eller den er signeret, men aldrig notariseret. Signér den med et Developer ID-certifikat med hardened runtime slået til og et sikkert tidsstempel, send den til notariseringstjenesten, og hæft billetten på, så Gatekeeper finder den, også uden netforbindelse. Et ZIP-arkiv kan notariseres, men kan ikke bære en billet, så hæft den på appen indeni:

```
ditto -c -k --keepParent App.app App.zip
xcrun notarytool submit App.zip --keychain-profile notary --wait
xcrun stapler staple App.app
```

Et containerimage kører på udviklerens Mac og dør på serveren med `exec format error`. Docker bygger som standard til platformen for den BuildKit-dæmon, der udfører bygget, og på en Mac med Apple silicon kører den dæmon i en virtuel arm64-maskine. En container deler værtens kerne, så uden emulering kører en x86-64-vært kun x86-64-kode. Tjek serveren med `uname -m`, og byg derefter til den med `docker build --platform linux/amd64`, eller byg i CI på serverens processorarkitektur.

Læs mere

- Mac User Guide — Apples egen vejledning til macOS, holdt ajour for hver aktuel udgivelse.
- Check if your Mac needs more RAM — grafen for belastning af hukommelse, og hvad dens tre farver betyder.
- Notarizing macOS software before distribution — Developer ID, hardened runtime, og hvordan Gatekeeper bruger billetten.
- Multi-platform builds — hvorfor arm64- og amd64-værter kræver hver sit image, og hvordan ét byg laver begge.
- container — Apples værktøj til at køre Linux-containere på en Mac, hver i sin egen lette virtuelle maskine.
- XNU — kildekoden til macOS' systemkerne med en kort forklaring på, hvordan Mach, FreeBSD og IOKit passer sammen.

Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv enheden eller tjenesten, de observerede symptomer og de data, der skal bevares. Vi kan fastlægge det sikreste nyttige undersøgelsesskridt.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er macOS?« 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/macos/