Engineer Programvare

Skrev en parser som verifiserer hvert FFI-kall

Skrev en parser som leser den faktiske C-headeren på 7 308 linjer og verifiserer hvert kallsted, hver enum-konstant og at hver klasse som eier en peker, er final, etter at en håndskrevet plassholder-header lot kall til tre fjernede funksjoner kompilere, lenke og krasje.

september 2026

7,308

linjer C-header, lest i stedet for antatt

Situasjon. Tidlig i prosjektet ble C-grensesnittet representert av en håndskrevet headerfil som beskrev funksjonene appen forventet. Headerfilen kompilerte, appen ble lenket, og kall til tre funksjoner som ikke lenger fantes i kjernen, kom helt frem til å bli utført og krasjet. Både kompilatoren og lenkeren hadde latt seg tilfredsstille av en beskrivelse av biblioteket i stedet for biblioteket selv, og gapet viste seg først under kjøring.

Oppgave. Den ekte headerfilen — 7 308 linjer og 268 deklarasjoner — måtte bli autoriteten, og hver bruk av den i Swift-koden måtte verifiseres mot den automatisk i stedet for ved gjennomgang.

Handling. Sjekken er en parser som leser den faktiske headerfilen fra oppstrøm og bygger mengden av funksjoner, enum-konstanter og typer den deklarerer, og deretter leser Swift-kildekoden og slår opp hvert kallsted og hver konstantreferanse mot den mengden. Et kall til en funksjon som headerfilen ikke deklarerer, får byggingen til å feile. En referanse til en enum-konstant som har fått nytt navn, får byggingen til å feile. Per i dag refereres det til 132 av de 268 deklarasjonene, og det er nyttig i seg selv å vite hvilke 136 som er ubrukt, fordi det sier nøyaktig hvor mye av kjernen klienten ikke har nådd. Parseren håndhever også en regel kompilatoren ikke kan håndheve: hver Swift-klasse som eier en peker inn i kjernen, må være final. En klasse som eier en peker og ikke er final,

kan få subklasser, og en subklasse som overstyrer deinitialiseringen eller legger til sin egen levetid, endrer når pekeren frigjøres — en use-after-free uten noe unsafe-nøkkelord i nærheten. Regelen sjekkes etter navn over hele treet.

Resultat. Den feiltypen som ga den opprinnelige krasjen, kan ikke komme tilbake, fordi en utdatert referanse nå er en byggefeil og ikke en kjøretidsfeil. Begrensningen er at parseren forstår headerfilens deklarasjoner og ikke semantikken: den beviser at det finnes en funksjon med et navn som stemmer, og en funksjon der betydningen eller eierskapsregelen er endret oppstrøms mens signaturen er beholdt, slipper gjennom uten kommentar.

Bruk erfaringen i prosjektet deres

Bruk «Skrev en parser som verifiserer hvert FFI-kall» som utgangspunkt for en lignende utfordring i organisasjonen deres. Tjenestene bak arbeidet er tilgjengelige for prosjektet deres. Delen brukerne aldri ser: hvor dataene ligger, hvilke regler som gjelder, og hvordan andre systemer får lov til å snakke med den.

Se tjenestene våre Ta kontakt →

Se bak resultatet

Les om situasjonen, beslutningene og resultatet i eksemplet. Vi kan drøfte hva som passer deres rammer, og hva som krever en annen tilnærming. Skrev en parser som verifiserer hvert FFI-kall — 7,308 linjer C-header, lest i stedet for antatt

Se tjenestene våre Ta kontakt →

Ville bygget ditt fange opp et kall til en funksjon som er borte?

Snakk om prosjektet med ingeniøren som har ansvaret for leveransen. Vi avtaler omfang, prisgrunnlag og overlevering skriftlig før arbeidet begynner.

Rolle i oppdraget: Applikasjonsingeniør for macOS. Det bygger på Backend-utvikling, API-er og integrasjon, Automatisering og CI/CD, Sikkerhet, Testing og kvalitetssikring og Dokumentasjon. Og nå som dere er her: vi er ganske gode på Backend- og API-utvikling, DevOps og CI/CD-automatisering og Teknisk dokumentasjon.

Ta kontakt Se tjenestene våre

Ett stykke av **en lengre oversikt** — og den ligger i sin helhet her.

Hver oppføring er skrevet på samme måte: situasjonen, oppgaven, hva vi gjorde, og hva som endret seg.

Se hele oversikten

Definer forbedringen dere trenger

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 →

Ville bygget ditt fange opp et kall til en funksjon som er borte?

Spør om beslutningene bak «Skrev en parser som verifiserer hvert FFI-kall». Dere snakker med folk som kan forklare arbeidet og vurdere deres eget prosjekt.

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://software.engineer.company/nb/portfolio/wrote-a-parser-verifying-every-ffi-call/