Engineer

Införde strikt samtidighet i Swift 6 ovanpå C

Införde fullständig strikt samtidighet i Swift 6 utan actors och överbryggade en blockerande händelseloop i C till main actor genom en producent, en konsument och en ordning — efter att ha konstaterat att en task per händelse förlorar den ordning som gränssnittet är beroende av.

september 2026

strikt samtidighet med en enda ordning, och inga actors

Situation. Den fullständiga strikta samtidighetskontrollen i Swift 6 gör datakapplöpningar till kompileringsfel i stället för sporadiska krascher. Det är när den införs mot ett C-bibliotek som det blir svårt: kärnans händelseloop är ett blockerande anrop som måste köras utanför huvudtråden för all framtid, och värdena den lämnar tillbaka är pekare utan några som helst samtidighetsgarantier. Kompilatorn kan inte resonera om något av det och vägrar allt tills gränsen är uttryckligen beskriven.

Uppgift. Fullständig kontroll måste slås på utan nödutgångar, vilket innebar att utforma övergången från en blockerande C-loop till main actor i stället för att annotera sig runt den.

Handling. Det självklara angreppssättet är en actor per delsystem, och det förkastades efter mätning snarare än av smakskäl. Att starta en task per inkommande händelse låter körmiljön schemalägga dem i godtycklig ordning, och kärnans händelseström är ordnad — en händelse om att ett meddelande har ändrats som går om händelsen om att samma meddelande har skapats ger ett gränssnitt som visar en redigering av något som ännu inte finns. Att actors är återinträdesbara gör saken värre, inte bättre, eftersom en actor kan pausa mitt i en metod och behandla ett annat anrop. Det som ersatte den är medvetet enkelt: en producenttråd som äger den blockerande loopen, en konsument, en kö mellan dem och ett enda hopp över till main actor på slutet. Ordningen bevaras eftersom det finns exakt en väg och ingenting går om något annat. De osäkra typer som korsar den gränsen kapslas in i typer vars trådsäkerhet intygas i

wrappern i stället för att antas, och intygandet är dokumenterat tillsammans med skälet till att det håller — pekaren ägs av en enda tråd och kopieras innan den lämnas över.

Resultat. Appen kompilerar med fullständig strikt samtidighetskontroll utan några undertryckta varningar, och händelseordningen är en strukturell egenskap snarare än en förhoppning. Priset är att designen är mindre parallell än den skulle kunna vara: allt kanaliseras genom en enda konsument, och om den konsumenten någon gång blir en flaskhals kräver lösningen att man på nytt härleder vilka händelser som säkert kan byta ordning, vilket är precis den analys som det här undvek.

Använd erfarenheten i ert projekt

Använd ”Införde strikt samtidighet i Swift 6 ovanpå C” som utgångspunkt för en liknande utmaning i er organisation. Tjänsterna bakom arbetet finns tillgängliga för ert projekt. Systemets form utarbetad på papper först, så att delarna passar ihop och kostnaden är känd innan någon skriver kod.

Se våra tjänster Hör av er →

Se bakom resultatet

Läs om situationen, besluten och resultatet i exemplet. Vi kan diskutera vad som passar era förutsättningar och vad som kräver ett annat angreppssätt. Införde strikt samtidighet i Swift 6 ovanpå C — strikt samtidighet med en enda ordning, och inga actors

Se våra tjänster Hör av er →

Vill du ha samtidighet som kompilatorn kan bevisa är säker?

Prata om ert projekt med ingenjören som ansvarar för leveransen. Vi avtalar omfattning, hur priset beräknas och överlämning skriftligt innan arbetet börjar.

Roll i uppdraget: Applikationsingenjör för macOS. Det bygger på Lösningsarkitektur, Backendutveckling, Frontendutveckling, Tillförlitlighet och säkerhetskopiering, Prestandaoptimering och Test och kvalitetssäkring. Och nu när ni är här: vi är ganska bra på Plattforms- och lösningsarkitektur, Backend- och API-utveckling och Tekniskt ledarskap och rådgivning.

Hör av er Se våra tjänster

En del av **en längre historik** — och den finns i sin helhet här.

Varje post är skriven på samma sätt: situationen, uppgiften, vad vi gjorde och vad som förändrades.

Se hela historiken

Definiera förbättringen ni behöver

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Vill du ha samtidighet som kompilatorn kan bevisa är säker?

Fråga om besluten bakom ”Införde strikt samtidighet i Swift 6 ovanpå C”. Ni talar med personer som kan förklara arbetet och bedöma ert eget projekt.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/portfolio/adopted-swift-6-strict-concurrency-over-c/