Engineer

Concurrence stricte de Swift 6 sur C

Adopté le mode complet de concurrence stricte de Swift 6 sans acteurs, en reliant une boucle d'événements C bloquante à l'acteur principal avec un producteur, un consommateur et un ordre unique, après avoir établi qu'une tâche par événement perdait l'ordre nécessaire à l'interface.

septembre 2026

concurrence stricte avec un ordre unique, sans acteurs

Situation. La vérification complète et stricte de la concurrence de Swift 6 transforme les courses de données en erreurs de compilation plutôt qu'en plantages intermittents. L'adopter face à une bibliothèque C est la partie difficile : la boucle d'événements du cœur est un appel bloquant qui doit rester indéfiniment hors du thread principal, et les valeurs qu'elle renvoie sont des pointeurs dépourvus de toute garantie de concurrence. Le compilateur ne peut raisonner sur rien de cela et refuse tout tant que la frontière n'est pas décrite explicitement.

Tâche. La vérification complète devait être activée sans échappatoire, ce qui exigeait de concevoir le passage d'une boucle C bloquante à l'acteur principal plutôt que de l'esquiver par des annotations.

Action. L'approche évidente consiste à utiliser un acteur par sous-système, et elle a été rejetée sur la base de mesures plutôt que de goûts. Créer une tâche par événement entrant laisse l'environnement d'exécution les ordonnancer dans n'importe quel ordre, alors que le flux d'événements du cœur est ordonné — un événement de modification de message qui dépasse l'événement de création auquel il se réfère produit une interface affichant une modification de quelque chose qui n'existe pas encore. La réentrance des acteurs aggrave cela au lieu de l'améliorer, car un acteur peut se suspendre au milieu d'une méthode et traiter un autre appel. Ce qui l'a remplacée est délibérément simple : un thread producteur possédant la boucle bloquante, un consommateur, une file entre eux et un seul passage vers l'acteur principal à la fin. L'ordre est préservé parce qu'il existe exactement un chemin et que rien ne dépasse rien. Les types non sûrs qui traversent cette frontière sont enveloppés dans des types

dont la sûreté multithread est affirmée au niveau de l'enveloppe plutôt que supposée, et cette affirmation est documentée avec sa justification — le pointeur appartient à un seul thread et est copié avant d'être transmis.

Résultat. L'application compile avec la vérification complète et stricte de la concurrence sans aucune suppression, et l'ordre des événements est une propriété structurelle plutôt qu'un espoir. Le coût est une conception moins parallèle qu'elle pourrait l'être : tout passe par un seul consommateur, et si celui-ci devient un goulet d'étranglement, la correction exigera de redéterminer quels événements peuvent être réordonnés sans risque, soit précisément l'analyse que cette conception a évitée.

Mettez cette expérience au service de votre projet

Prenez « Concurrence stricte de Swift 6 sur C » comme point de départ pour un défi similaire dans votre organisation. Les prestations mobilisées pour cette réalisation sont disponibles pour votre projet. Définir d'abord le système sur le papier pour que ses éléments s'articulent et que son coût soit connu avant d'écrire du code.

Découvrir nos prestations Contactez-nous →

Examinez ce qui a conduit au résultat

Découvrez la situation, les décisions et le résultat de cet exemple. Nous pouvons discuter des éléments transposables à vos contraintes et de ceux qui nécessitent une autre démarche. Concurrence stricte de Swift 6 sur C — concurrence stricte avec un ordre unique, sans acteurs

Découvrir nos prestations Contactez-nous →

Envie d'une concurrence dont le compilateur peut prouver la sûreté ?

Discutez de votre projet avec l'ingénieure responsable de sa réalisation. Nous convenons par écrit du périmètre, de la base tarifaire et de la transmission avant le début des travaux.

Rôle exercé dans ce contexte : Ingénieure applications macOS. Cette réalisation fait appel à Architecture de solutions, Ingénierie backend, Ingénierie frontend, Fiabilité et sauvegardes, Optimisation des performances et Tests et assurance qualité. Et puisque vous êtes ici, nous sommes aussi plutôt doués en Architecture de plateformes et de solutions, Développement backend et API et Direction technique et conseil.

Contactez-nous Découvrir nos prestations

Un élément **d'un parcours plus long** —
entièrement présenté sur ce site.

Chaque entrée suit la même structure : la situation, la tâche, ce que nous avons fait et ce qui a changé.

Voir l'ensemble du parcours

Définissez l'amélioration nécessaire

Présentez le processus de travail, les interfaces existantes et les utilisateurs qui ont besoin du changement. Nous pouvons définir une livraison testable et une transmission permettant à votre équipe d'assurer la maintenance des éléments livrés.

Consulter les tarifs Contactez-nous →

Envie d'une concurrence dont le compilateur peut prouver la sûreté ?

Interrogez-nous sur les décisions qui sous-tendent « Concurrence stricte de Swift 6 sur C ». Vous parlerez aux personnes capables d'expliquer le travail et d'évaluer votre propre projet.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

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

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/portfolio/adopted-swift-6-strict-concurrency-over-c/