

実演でわかる！

async/await や actor の
iOS アプリへの取り入れ方

Yuta Koshizawa @koher

Xcode 13.2

- 今週リリースされた
- Swift Concurrency の back deployment
 - iOS 13 以降で `async/await` や `actor` が使えるように¹

¹ https://developer.apple.com/documentation/xcode-release-notes/xcode-13_2-release-notes



Use Swift Concurrency #265

Changes from all commits

File filter

Conversations

Jump to



0 / 30 files viewed



Review changes

53 Shared/Repository/Monsters/MonstersRepository.swift

Viewed



```
9
10 /// @mockable
11 protocol MonstersRepository: AnyObject { // swiftlint:disable:this file_types_order
12 - func loadMonsters(_ completion: @escaping (Result<MonsterDT0, Error>) -> Void)
13 }
```

```
14
15 final class MonstersFirebaseClient {
```

```
@@ -18,35 +18,32 @@
```

```
18
19 extension MonstersFirebaseClient: MonstersRepository {
20
21 - func loadMonsters(_ completion: @escaping (Result<MonsterDT0, Error>) -> Void) {
22 -     let monstersRef = self.firestore.collection("monsters")
23 -     monstersRef.getDocuments { querySnapshot, error in
24 -         if let error = error {
25 -             completion(.failure(error))
26 -             return
27         }
28     }
29 - guard let querySnapshot = querySnapshot else {
30 -     fatalError("Fail to unwrap `querySnapshot`.")
31 - }
```

```
27     }
28
29 - guard let querySnapshot = querySnapshot else {
30 -     fatalError("Fail to unwrap `querySnapshot`.")
31 - }
```

```
27     }
28
29 - guard let querySnapshot = querySnapshot else {
30 -     fatalError("Fail to unwrap `querySnapshot`.")
31 - }
```

```
9
10 /// @mockable
11 protocol MonstersRepository: AnyObject { // swiftlint:disable:this file_types_order
12 + func loadMonsters() async throws -> [MonsterDT0]
13 }
```

```
14
15 final class MonstersFirebaseClient {
```

```
@@ -18,35 +18,32 @@
```

```
18
19 extension MonstersFirebaseClient: MonstersRepository {
20
21 + func loadMonsters() async throws -> [MonsterDT0] {
22 +     let monstersRef = firestore.collection("monsters")
23 +     let querySnapshot = try await monstersRef.getDocuments()
24 +
25 +     var monsters: [MonsterDT0] = []
26 +     for document in querySnapshot.documents.filter({ $0.exists }) {
27 +         let monster = document.data()
28 +         guard let name = monster["name"] as? String,
29 +             let description = monster["description"] as? String,
30 +             let baseColorCode = monster["base_color"] as? String,
31 +             let iconUrlString = monster["icon_url"] as? String,
32 +             let dancingUrlString = monster["dancing_url"] as? String,
33 +             let order = monster["order"] as? Int else {
34 +             continue
35         }
36     }
37 +     monsters.append(MonsterDT0(
38 +         name: name,
39 +         description: description,
```

```
35     }
36
37 +     monsters.append(MonsterDT0(
38 +         name: name,
39 +         description: description,
```

Check warning on line 34 in Shared/Repository/Monsters/MonstersRepository.swift



Codecov / codecov/patch

Shared/Repository/Monsters/MonstersRepository.swift#L34

Added line #L34 was not covered by tests

² <https://github.com/uhooi/UhooiPicBook/pull/265/files>

今年の振り返り

- Swift Zoomin' #5: 先取り！ Swift 6 の `async/await` （1月）
- `async/await` や `actor` で iOS アプリ開発がどう変わるか
Before & After の具体例で学ぶ （9月）
- Swift Zoomin' #8: Actor ハンズオン （10月）

async/await や actor を
どうやって iOS アプリに取り入れるか

実際に見るのが一番早い

ライブコーディング

今日扱うアプリ

- API を叩いてユーザーの JSON を取得
 - <https://koherent.org/fake-service/api/user?id=1234>
- JSON をデコードしてユーザー情報を表示
- JSON の中に記載された URL からアイコンをダウンロード
- ユーザーのアイコンも表示

今日のポイント

- Other Swift Flags に `-Xfrontend -warn-concurrency`
- コールバックから `async` への変換には `Continuation`
- 非同期関数を始めるところでは `Task.init`
 - SwiftUI & iOS 15 以降では `.task`³

³ <https://developer.apple.com/videos/play/wwdc2021/10019/>

今日のポイント

- メインスレッドで実行するときは `MainActor.run`
- Actor 境界を越える型は `Sendable` に適合させる
- `ObservableObject` には `@MainActor` を付与
- `View Controller` や `View` には `@MainActor` を付与