# Как адаптировать проект к Complete Concurrency Checking

## Ожидание

**Race condition**
**Невозможен**

## Ожидание

**Race condition**
**Невозможен**

## Реальность

**Переменная**
**это**
**Ошибка**

# Май 2024

Начало проекта

🔬 Все модули не изолированны

✅ Включили Complete Concurrency Checking на весь проект

❌ Проект не собирается, имеем более 1к ошибок и ворнингов

🧠 Добавили per module возможность включить Complete Concurrency Checking

# Май 2024

Начало проекта

🔬 Все модули не изолированны

✅ Включили Complete Concurrency Checking на весь проект

❌ Проект не собирается, имеем более 1к ошибок и ворнингов

🧠 Добавили per module возможность включить Complete Concurrency Checking

# Май 2024

Начало проекта

🔬 Все модули не изолированны

✅ Включили Complete Concurrency Checking на весь проект

❌ Проект не собирается, имеем более 1к ошибок и ворнингов

🧠 Добавили per module возможность включить Complete Concurrency Checking

# Май 2024

Начало проекта

🔬 Все модули не изолированны

✅ Включили Complete Concurrency Checking на весь проект

❌ Проект не собирается, имеем более 1к ошибок и ворнингов

🧠 Добавили per module возможность включить Complete Concurrency Checking

# Начинаем копать

## 🛠️ Основные инструменты

**Actors**

@MainActor

@globalActor

Sendable

Sendable из коробки

@unchecked Sendable

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

# Actors

# Actors

## Можем аннотировать:

- **Тип**
- Проперти
- Метод


# Actors

## Можем аннотировать:

- Тип
- **Проперти**
- Метод

FirstSample.swift | SecondSample.swift | ThirdSample.swift

```
public final class SecondSample {
  @MainActor
  public var intArray = [Int]()
}
```

# Actors

## Можем аннотировать:

- Тип
- Проперти
- **Метод**

FirstSample.swift SecondSample.swift ThirdSample.swift

```
public final class ThirdSample {
  @MainActor
  public func doStuff() {...}
}
```

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| **@MainActor** | Sendable из коробки |
| @globalActor | @unchecked Sendable |

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

# MainActor

```
@MainActor
private func deselectPins() {
  guard let map = mapObject?.mapWindow.map else {
    return
  }
  map.deselectGeoObject()
}
```

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| @MainActor | Sendable из коробки |
| **@globalActor** | @unchecked Sendable |

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

# GlobalActor

```
@globalActor
public actor MediaManagerActor {
  public static let shared = MediaManagerActor()
}
```

# GlobalActor

```
@MediaManagerActor
public final class MediaManager {

  public nonisolated init() {}

  public func resizeUIImageToDataIfNeeded(
    _ image: UIImage,
    maxSize: MediaSize? = nil
  ) -> Data? {...}

  private func applyOrientationTransform(
    for ciImage: CIImage,
    orientation: UIImage.Orientation
  ) -> CIImage {...}
}
```

# GlobalActor

```
@MediaManagerActor
public final class MediaManager {

  public nonisolated init() {}

  public func resizeUIImageToDataIfNeeded(
    _ image: UIImage,
    maxSize: MediaSize? = nil
  ) -> Data? {...}

  private func applyOrientationTransform(
    for ciImage: CIImage,
    orientation: UIImage.Orientation
  ) -> CIImage {...}
}
```

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| @MainActor | Sendable из коробки |
| @globalActor | @unchecked Sendable |

## 🗜️ Utility

@preconcurency Import

**nonisolated**

**nonisolated(unsafe)**

# Nonisolated

```
@MainActor
public final class HotelDetailsStore {
  let snackbarStateMachine: SnackbarStateMachine

  public nonisolated init(
    snackbarStateMachine: SnackbarStateMachine
  ) {
    self.snackbarStateMachine = snackbarStateMachine
    Main actor-isolated property 'snackbarStateMachine' can not
    be mutated from a non-isolated context
  }
}
```

```
public final class SnackbarStateMachine {...}
```

# Nonisolated

```
@MainActor
public final class HotelDetailsStore {
  let snackbarStateMachine: SnackbarStateMachine

  public nonisolated init(
    snackbarStateMachine: SnackbarStateMachine
  ) {
    self.snackbarStateMachine = snackbarStateMachine
  }
}
```

```
public final class SnackbarStateMachine: Sendable {...}
```

# Nonisolated(unsafe)

```
public enum CurrencyFormatter {
  private static var formatters = [String: NumberFormatter]()
  private static var formatterLock = NSLock()

  public static func reset() {
    formatterLock.lock()
    defer { formatterLock.unlock() }
    formatters = [:]
  }

  private static func createFormatter(
    with code: String,
    withFractionDigits: Bool
  ) -> NumberFormatter {...}
```

Reference to static property 'formatterLock' is not concurrency-safe because it involves shared mutable state

# Nonisolated(unsafe)

```
public enum CurrencyFormatter {
  private nonisolated(unsafe) static var formatters = [String: NumberFormatter]()
  private nonisolated(unsafe) static var formatterLock = NSLock()

  public static func reset() {
    formatterLock.lock()
    defer { formatterLock.unlock() }
    formatters = [:]
  }

  private static func createFormatter(
    with code: String,
    withFractionDigits: Bool
  ) -> NumberFormatter {...}
```

# Nonisolated

```
@MainActor
public final class HotelDetailsStore {
  let snackbarStateMachine: SnackbarStateMachine

  public nonisolated init(
    snackbarStateMachine: SnackbarStateMachine
  ) {
    self.snackbarStateMachine = snackbarStateMachine
  }
}
```

```
public final class SnackbarStateMachine: Sendable {...}
```

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | **Sendable** |
| @MainActor | Sendable из коробки |
| @globalActor | @unchecked Sendable |

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

# Sendable

```
public final class SendableExample {
  var intArray = [Int]()

  func doStuff(){
    Task{
      await NetworkStuffDoer.doNetworkStuff(intArray)
      Capture of 'self' with non-sendable type 'SendableExample' in a `@Sendable` closure
    }
  }
}
```

# Sendable

```
public final class SendableExample: Sendable {
  var intArray = [Int]()
  Stored property 'intArray' of 'Sendable'-conforming class 'SendableExample' is mutable

  func doStuff(){
    Task{
      await NetworkStuffDoer.doNetworkStuff(intArray)
    }
  }
}
```

## Ожидание

**Race condition**
**Невозможен**

## Реальность

**Переменная**
**это**
**Ошибка**

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| @MainActor | Sendable из коробки |
| @globalActor | **@unchecked Sendable** |

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

# @unchecked Sendable

SendableExample.swift | Atomic.swift

```
public final class SendableExample: @unchecked Sendable {
  @Atomic var intArray = [Int]()

  func doStuff(){
    Task {
      await NetworkStuffDoer.DoNetworkStuff(intArray)
    }
  }
}
```

# @unchecked Sendable

SendableExample.swift | Atomic.swift

```swift
@propertyWrapper
public final class Atomic<Value> {
  public var wrappedValue: Value {
    get {
      lock.read {
        value
      }
    }
    set {
      lock.write {
        value = newValue
      }
    }
  }

  private var value: Value
  private let lock = Lock()

  public init(wrappedValue: Value) {
    value = wrappedValue
  }
}
```

# @unchecked Sendable

SendableExample.swift | Atomic.swift

```swift
public final class SendableExample: @unchecked Sendable {
  @Atomic var intArray = [Int]()

  var mutableState: Int = .zero

  func doStuff(){
    Task {
      await NetworkStuffDoer.DoNetworkStuff(intArray)
    }
  }
}
```

# @unchecked Sendable

SendableExample.swift | Atomic.swift

```swift
public final class SendableExample: @unchecked Sendable {
  @Atomic var intArray = [Int]()

  var mutableState: Int = .zero

  func doStuff(){
    Task {
      await NetworkStuffDoer.DoNetworkStuff(intArray)
    }
  }
}
```

# ThreadSafeMacro

SendableExample.swift

```swift
import ThreadSafeMacro

public final class SendableExample: Sendable {
  @ThreadSafe var intArray = [Int]()

  var mutableState: Int = .zero

  func doStuff(){
    Task {
      await NetworkStuffDoer.DoNetworkStuff(intArray)
    }
  }
}
```

# ThreadSafeMacro

```
private let _intArray = ThreadSafeStorage<[Int]>([])
{
    _read {
        yield _intArray.wrappedValue
    }
    _modify {
        yield &_intArray.wrappedValue
    }
}
```

# ThreadSafeMacro

SendableExample.swift

```swift
import ThreadSafeMacro

public final class SendableExample: Sendable {
  @ThreadSafe var intArray = [Int]()

  var mutableState: Int = .zero

  func doStuff(){
    Task {
      await NetworkStuffDoer.DoNetworkStuff(intArray)
    }
  }
}
```

Stored property 'mutableState' of 'Sendable'-conforming class 'SendableExample' is mutable

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| @MainActor | **Sendable из коробки** |
| @globalActor | @unchecked Sendable |

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

# Sendable из коробки

```
internal class Person {
  let name: String
  let lastName: String

  init(name: String, lastName: String) {
    self.name = name
    self.lastName = lastName
  }
}
```

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| @MainActor | Sendable из коробки |
| @globalActor | @unchecked Sendable |

## 🗜️ Utility

**@preconcurency Import**

nonisolated

nonisolated(unsafe)

# Preconcurrency

# Начинаем копать

## 🛠️ Основные инструменты

| | |
|---|---|
| Actors | Sendable |
| @MainActor | Sendable из коробки |
| @globalActor | @unchecked Sendable |

## 🗜️ Utility

@preconcurency Import

nonisolated

nonisolated(unsafe)

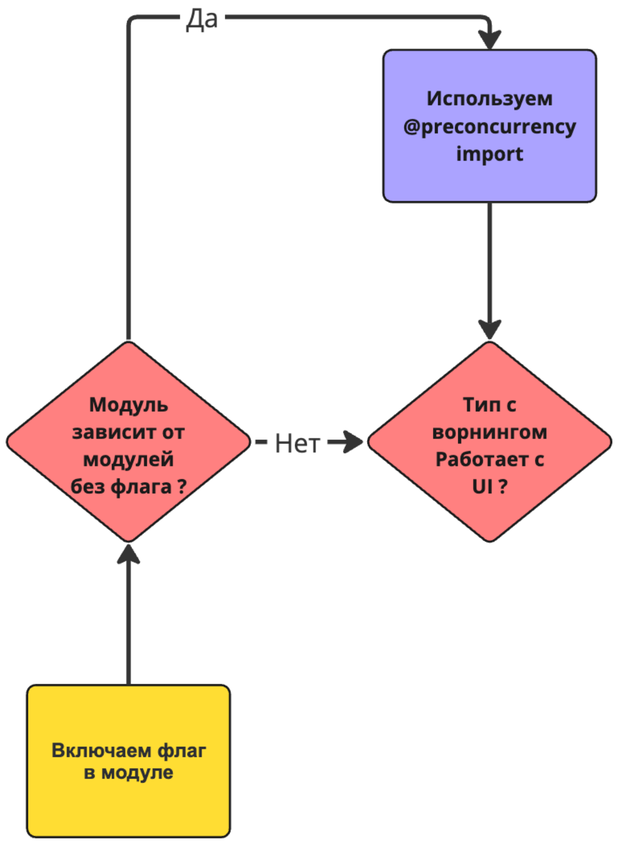
# Решение

# Решение

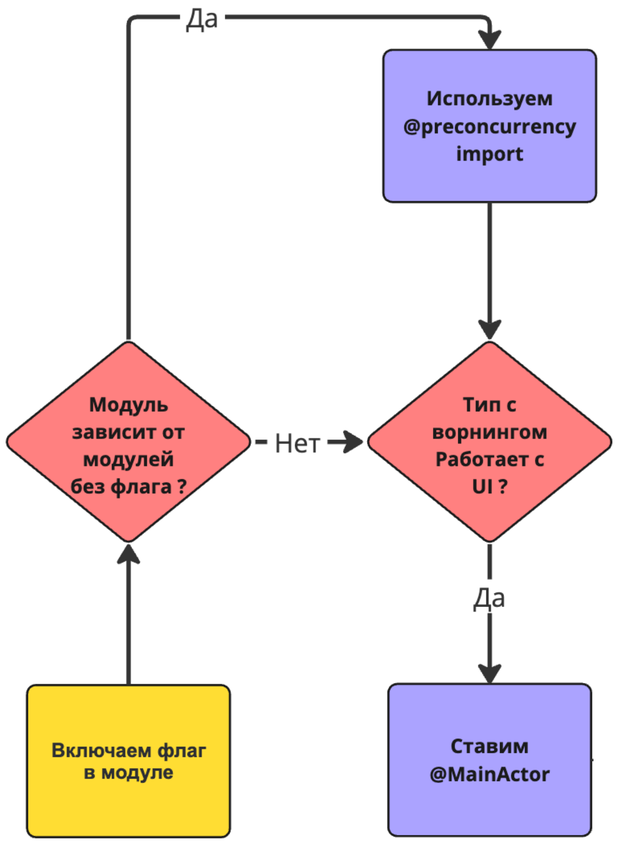
# Решение

Включаем флаг
в модуле

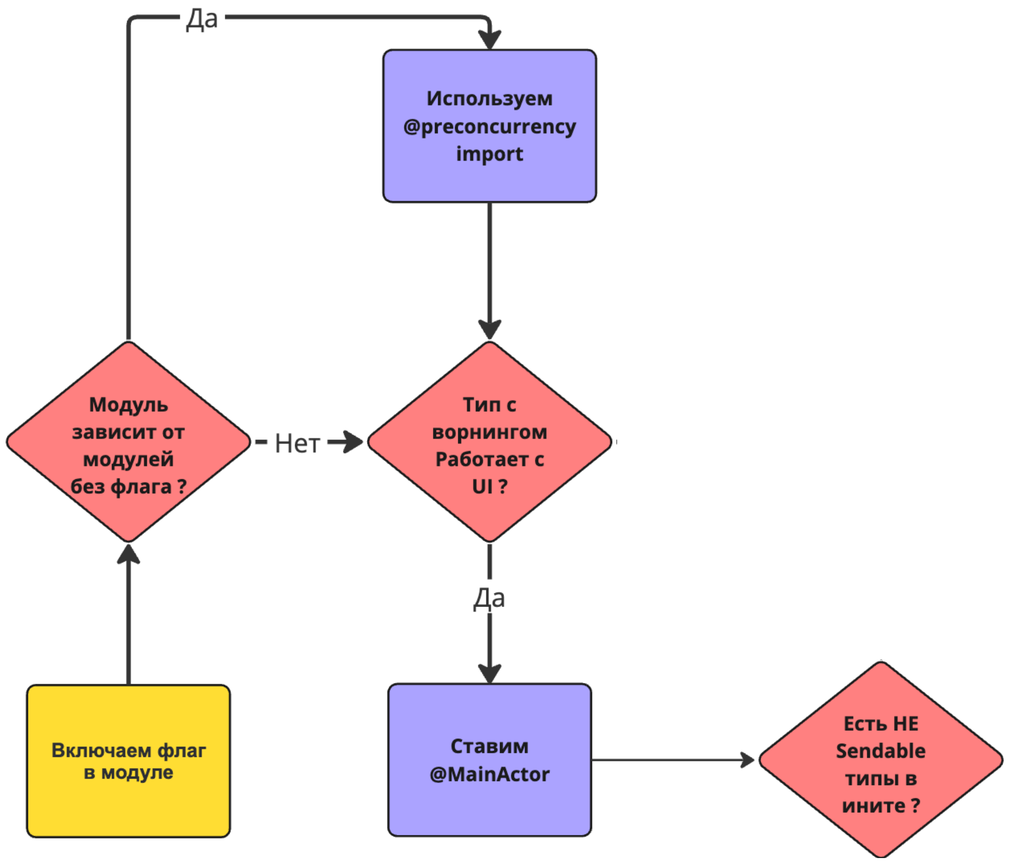
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Включаем флаг в модуле

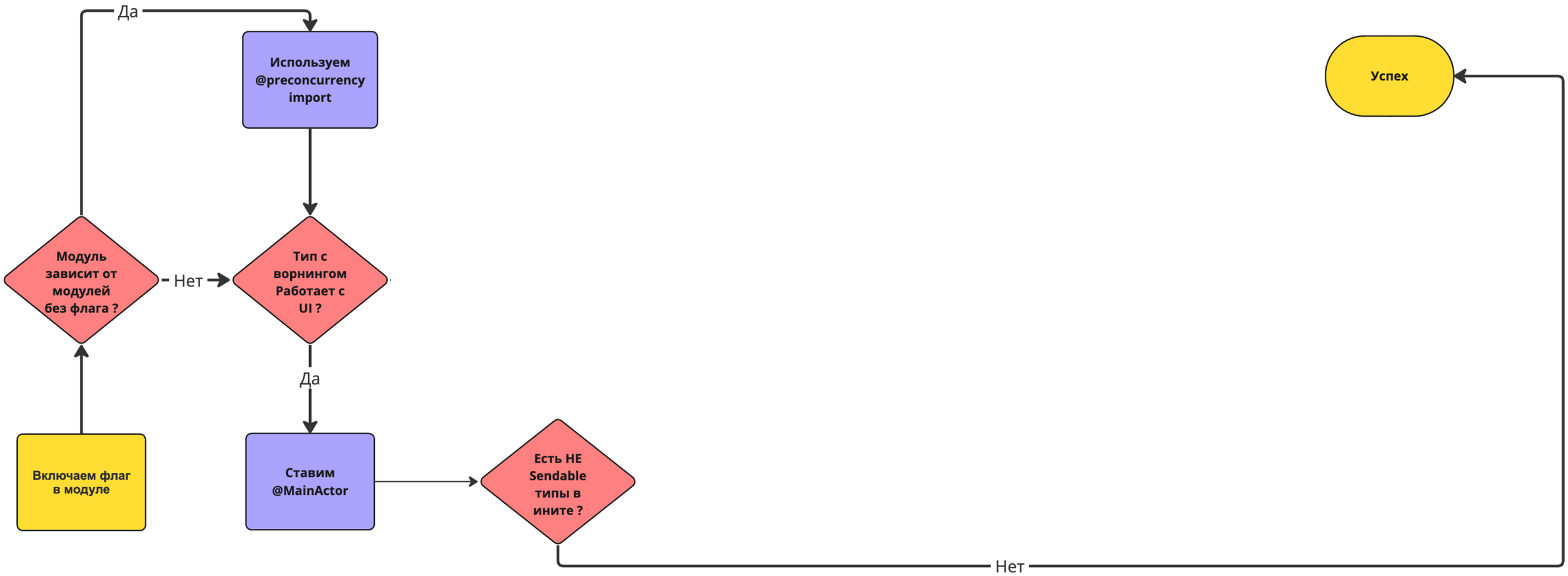
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor

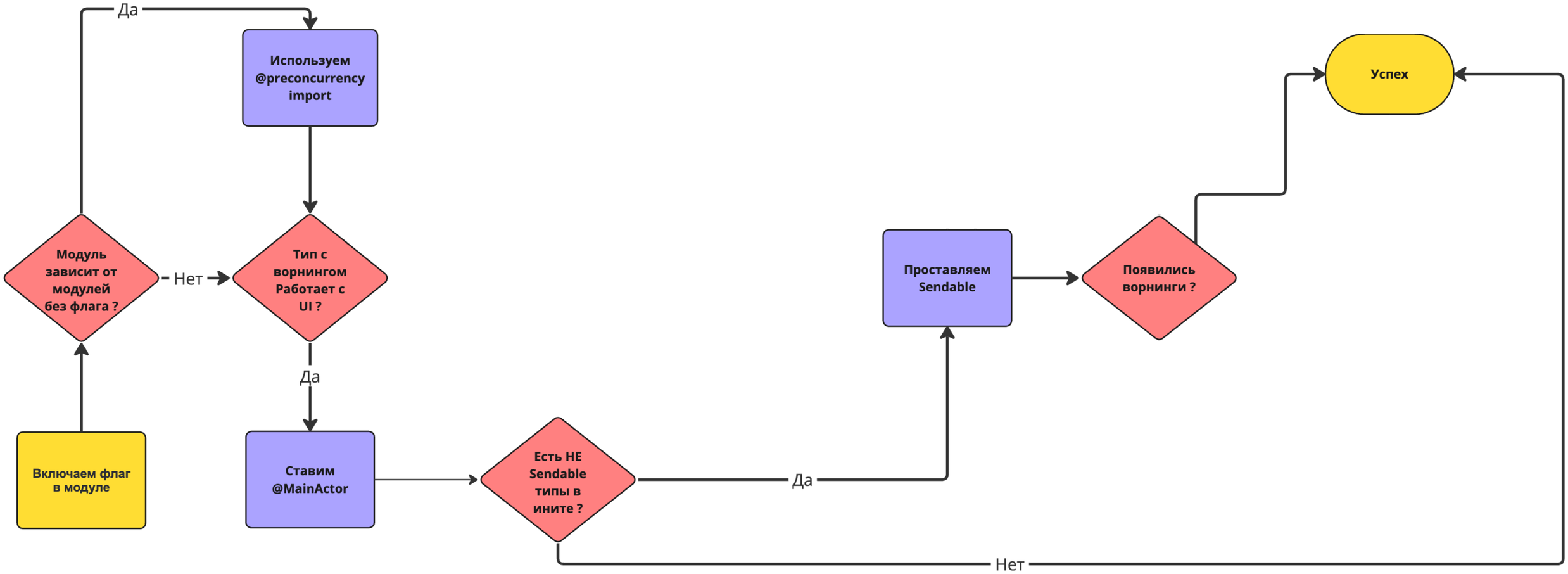
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?

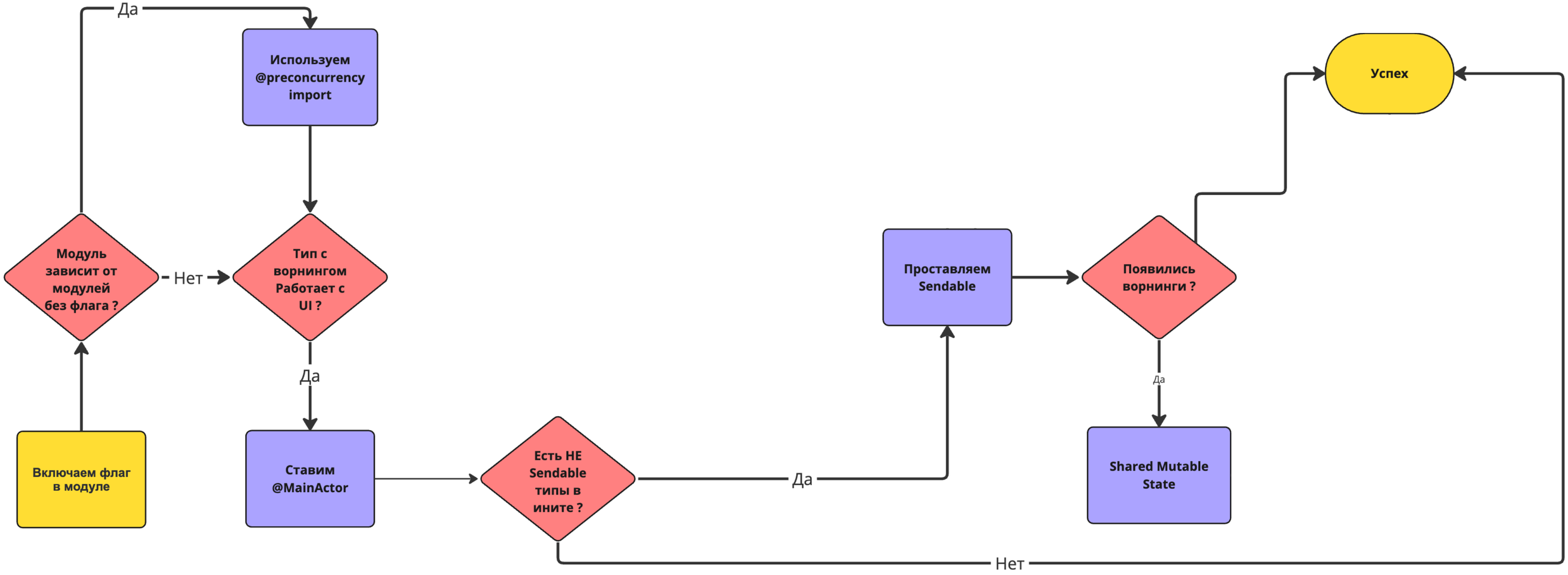
# Решение


Да
Используем @preconcurrency import
Успех
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Нет

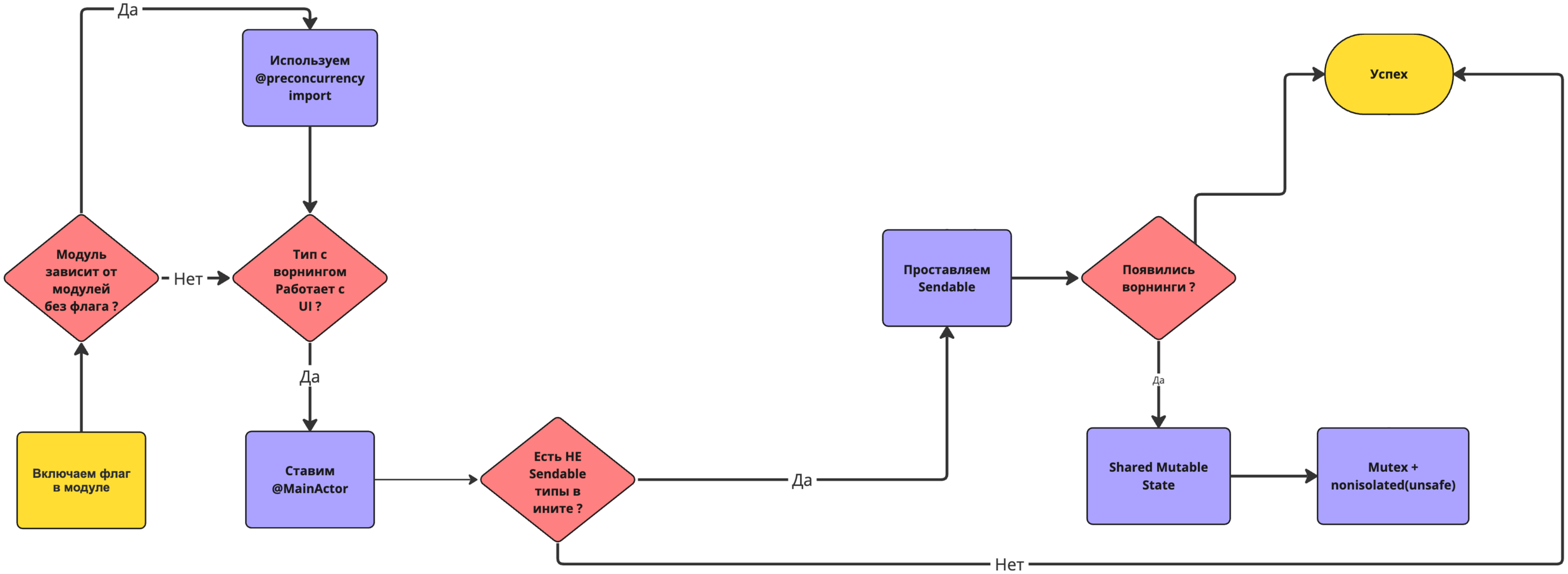
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Проставляем Sendable
Появились ворнинги ?
Успех
Нет

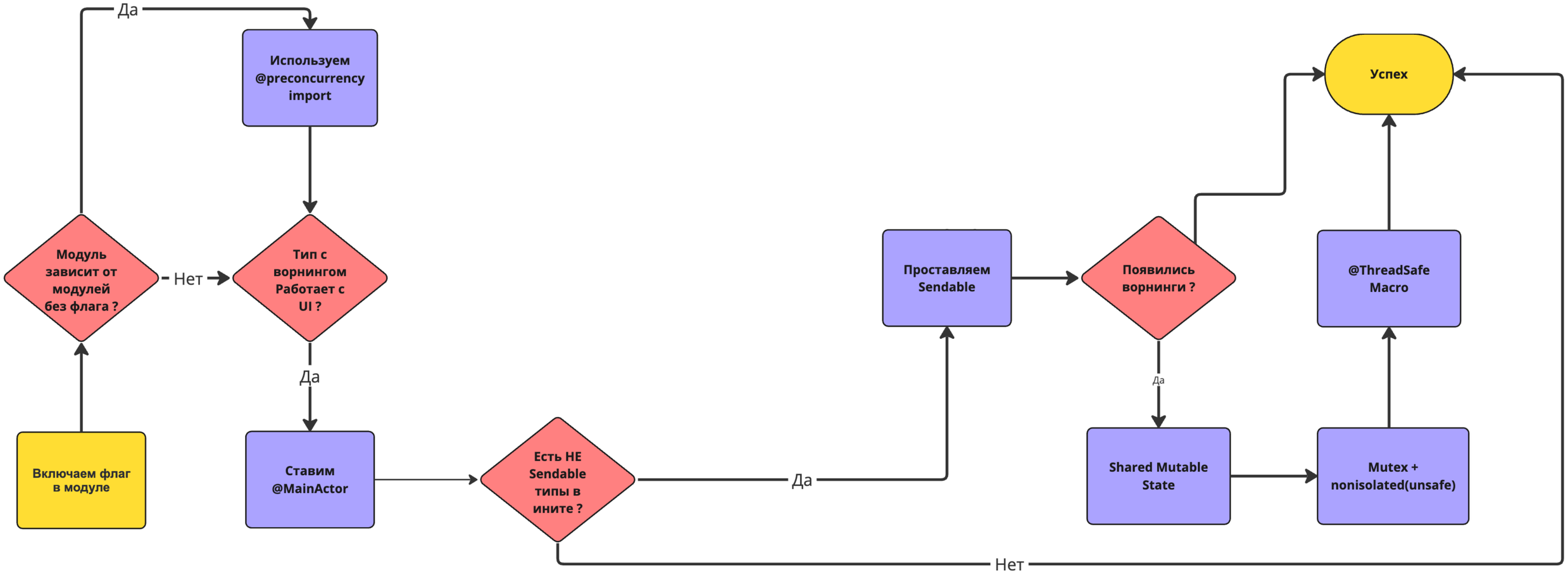
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Проставляем Sendable
Появились ворнинги ?
Да
Успех
Shared Mutable State
Нет

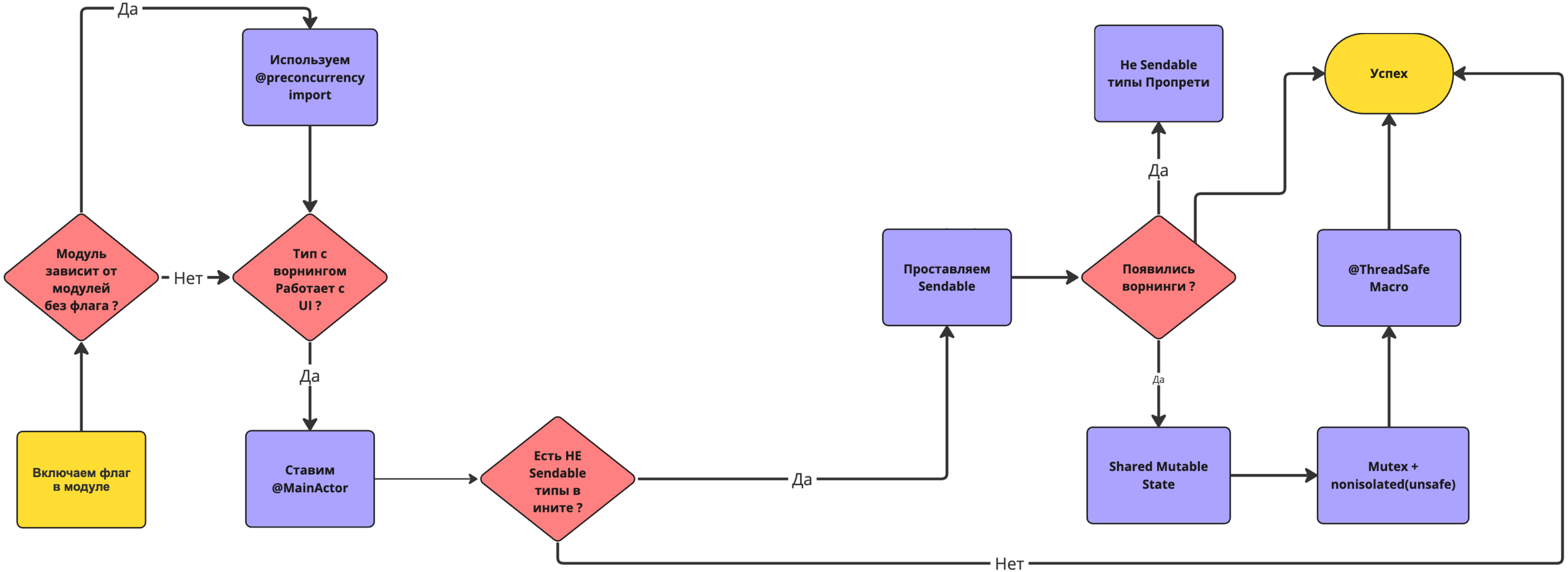
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Проставляем Sendable
Появились ворнинги ?
Да
Успех
Shared Mutable State
Mutex + nonisolated(unsafe)
Нет

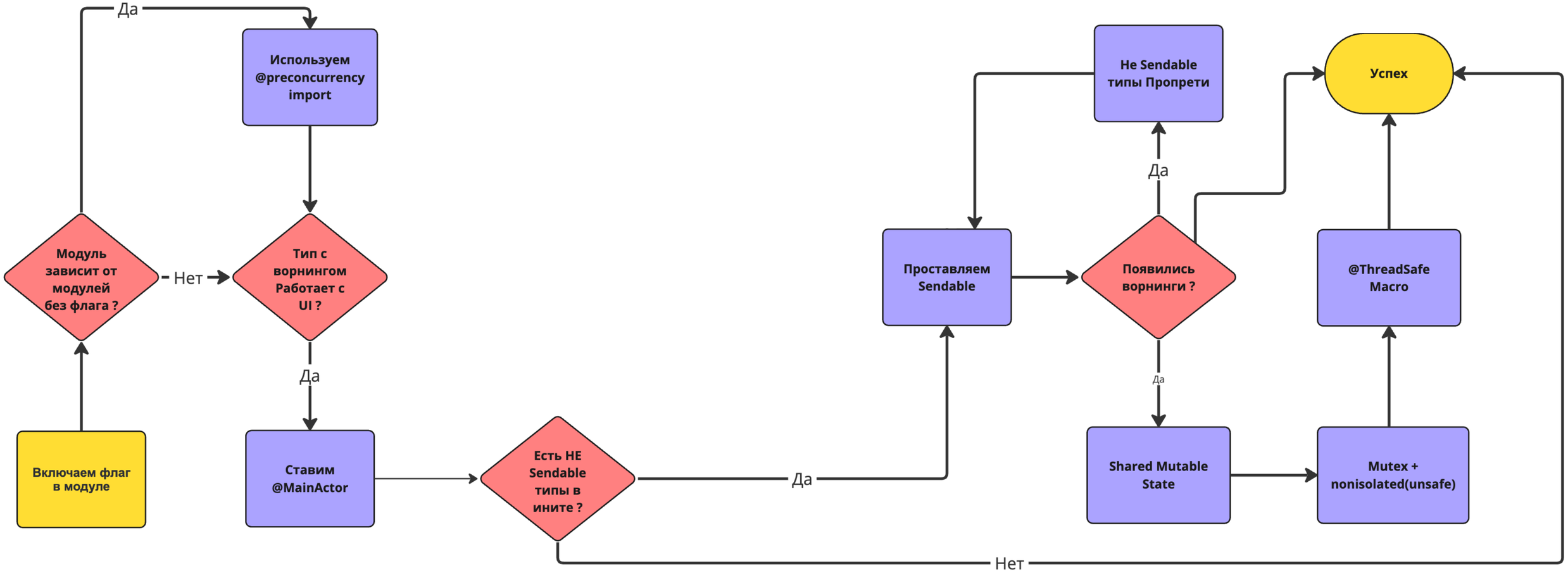
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Проставляем Sendable
Появились ворнинги ?
Да
Shared Mutable State
Mutex + nonisolated(unsafe)
@ThreadSafe Macro
Успех
Нет

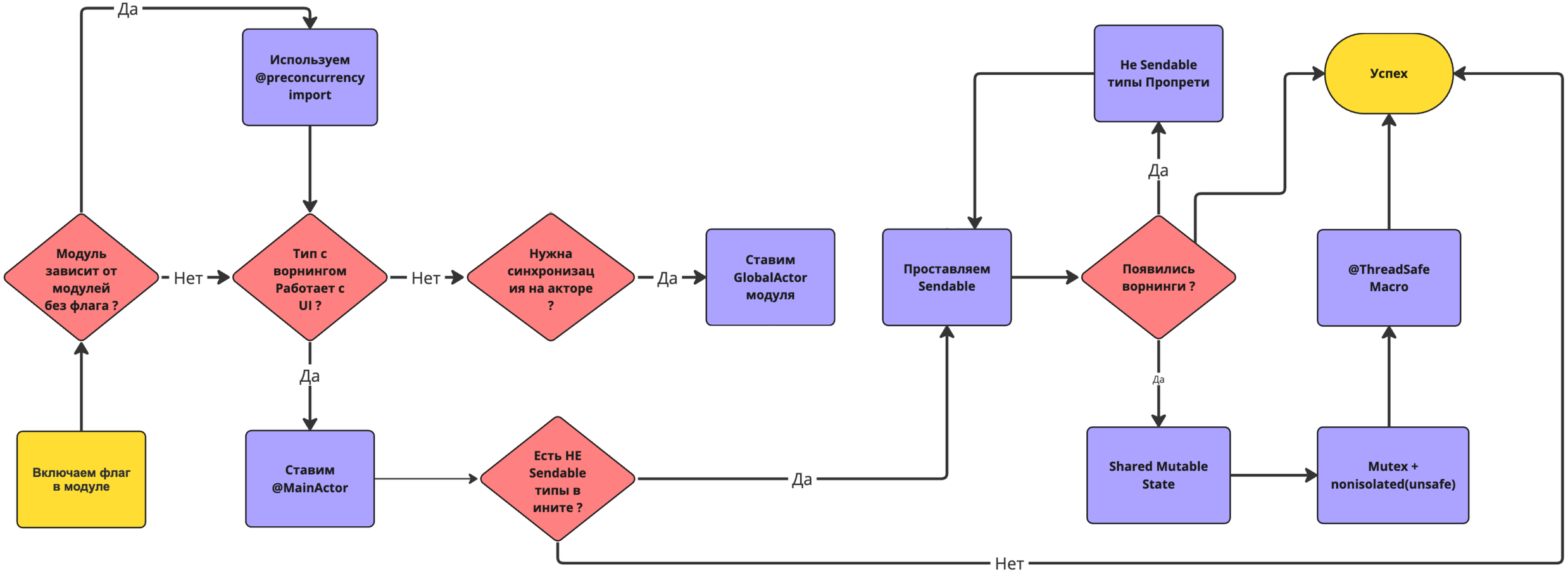
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Проставляем Sendable
Появились ворнинги ?
Да
Не Sendable типы Пропрети
Да
Shared Mutable State
Mutex + nonisolated(unsafe)
@ThreadSafe Macro
Успех
Нет

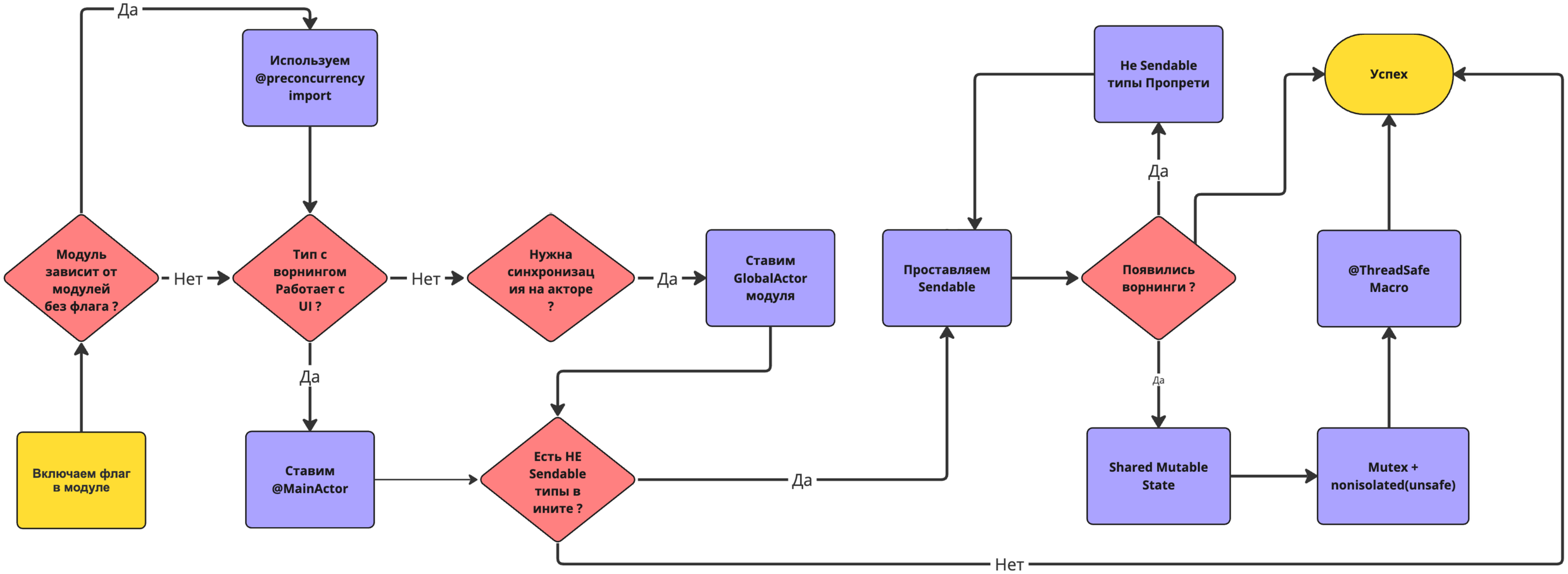
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в ините ?
Да
Проставляем Sendable
Появились ворнинги ?
Да
Да
Не Sendable типы Пропрети
Shared Mutable State
@ThreadSafe Macro
Успех
Mutex + nonisolated(unsafe)
Нет

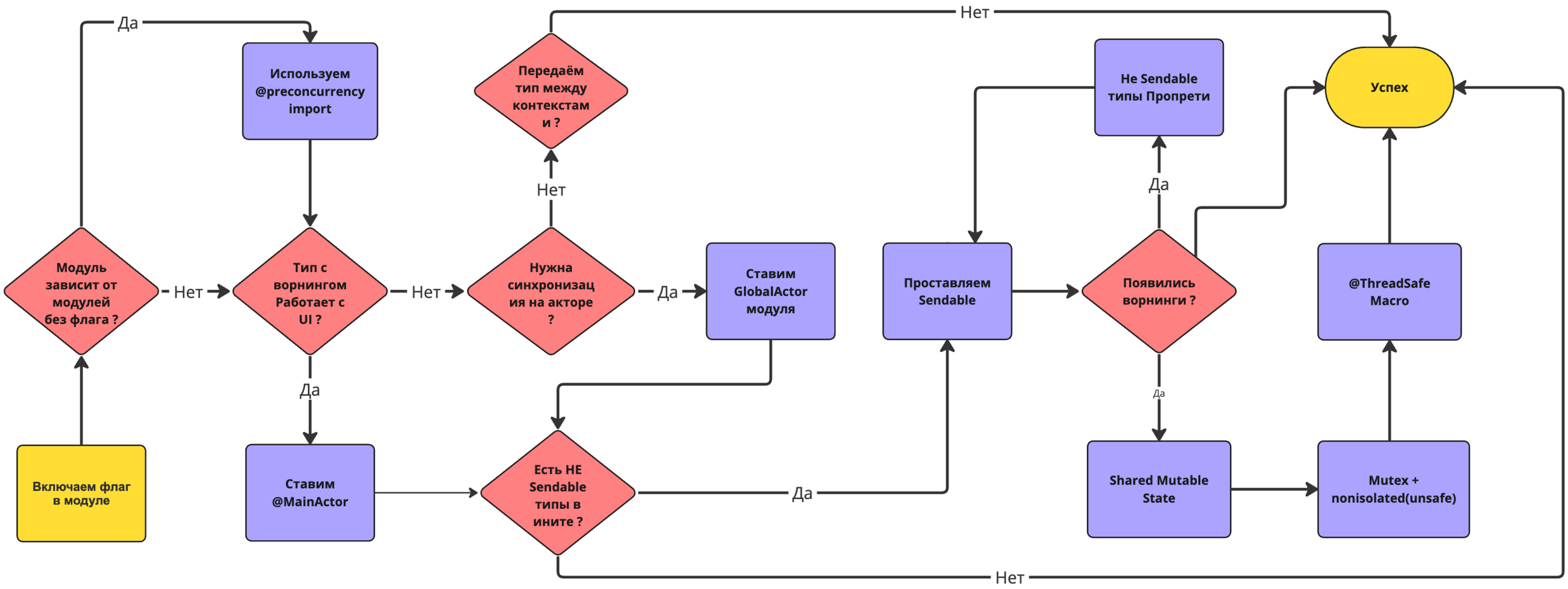
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Нет
Нужна синхронизация на акторе ?
Да
Ставим GlobalActor модуля
Проставляем Sendable
Не Sendable типы Пропрети
Появились ворнинги ?
Да
Успех
@ThreadSafe Macro
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Shared Mutable State
Mutex + nonisolated(unsafe)
Нет

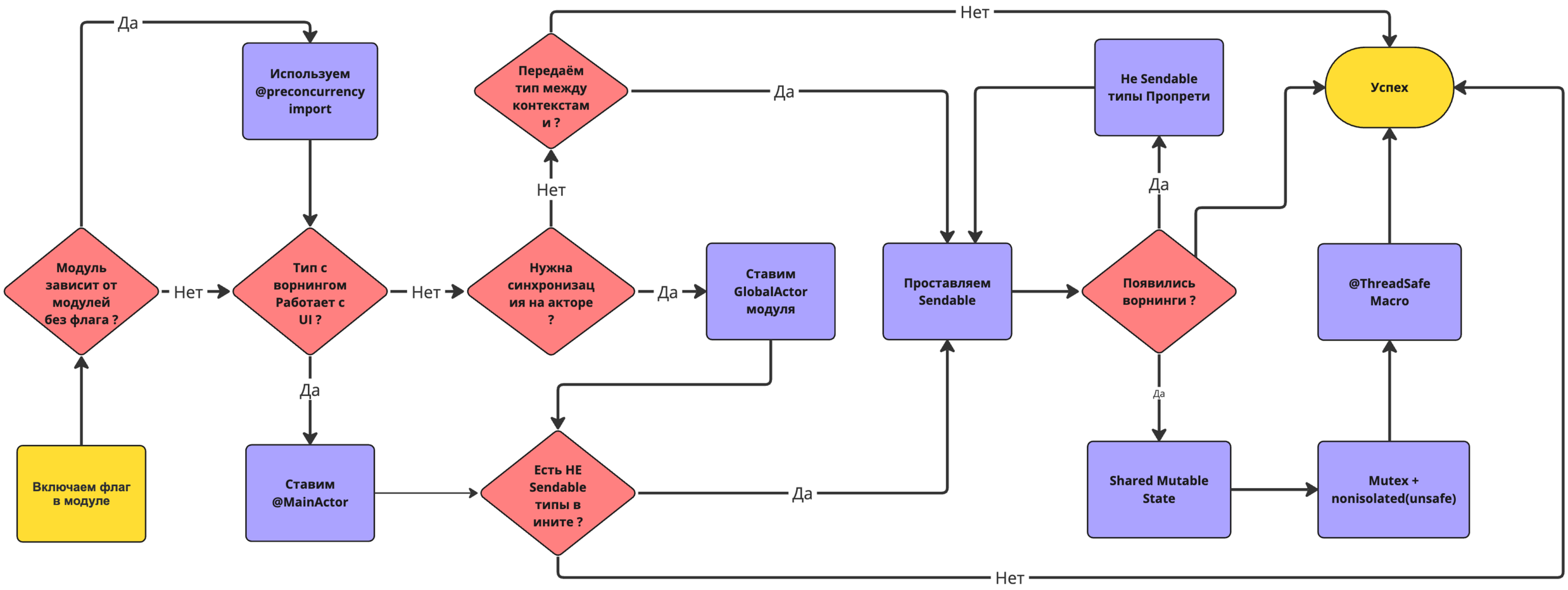
# Решение


Да
Используем @preconcurrency import
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Нет
Нужна синхронизация на акторе ?
Да
Ставим GlobalActor модуля
Проставляем Sendable
Не Sendable типы Пропрети
Появились ворнинги ?
Да
Успех
@ThreadSafe Macro
Включаем флаг в модуле
Да
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Да
Shared Mutable State
Mutex + nonisolated(unsafe)
Нет

# Решение


Да
Нет
Используем @preconcurrency import
Передаём тип между контекстам и ?
Нет
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Нет
Нужна синхронизац ия на акторе ?
Да
Ставим GlobalActor модуля
Проставляем Sendable
Не Sendable типы Пропрети
Да
Появились ворнинги ?
Успех
@ThreadSafe Macro
Включаем флаг в модуле
Да
Ставим @MainActor
Есть НЕ Sendable типы в ините ?
Да
Да
Shared Mutable State
Mutex + nonisolated(unsafe)
Нет

# Решение


Да
Нет
Используем @preconcurrency import
Передаём тип между контекстами ?
Да
Нет
Не Sendable типы Пропрети
Успех
Модуль зависит от модулей без флага ?
Нет
Тип с ворнингом Работает с UI ?
Нет
Нужна синхронизация на акторе ?
Да
Ставим GlobalActor модуля
Проставляем Sendable
Появились ворнинги ?
Да
@ThreadSafe Macro
Да
Да
Включаем флаг в модуле
Ставим @MainActor
Есть НЕ Sendable типы в инite ?
Да
Shared Mutable State
Mutex + nonisolated(unsafe)
Нет

# Внедрение

## если у вас многомодульный проект мало / нету GCD

- [ ] Есть способы per module включение флага
- [ ] Включаем флаг по умолчанию, на старых модулях выключаем
- [ ] Поддерживаем флаг в простых модулях
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## мало / нету GCD

- [ ] Есть способы per module включение флага
- [ ] Включаем флаг по умолчанию, на старых модулях выключаем
- [ ] Поддерживаем флаг в простых модулях
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## мало / нету GCD

- [x] Есть способы per module включение флага
- [ ] Включаем флаг по умолчанию, на старых модулях выключаем
- [ ] Поддерживаем флаг в простых модулях
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## мало / нету GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [ ] Поддерживаем флаг в простых модулях
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## мало / нету GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## мало / нету GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [x] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## мало / нету GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [x] Создаём задачи по изоляции сложных модулей
- [x] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## есть GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [ ] Модули с GCD обмазываем @unchecked Sendable
- [ ] Нарезаем их рефакторинг на маленькие задачи
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## есть GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [ ] Модули с GCD обмазываем @unchecked Sendable
- [ ] Нарезаем их рефакторинг на маленькие задачи
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект
## есть GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [x] Модули с GCD обмазываем @unchecked Sendable
- [ ] Нарезаем их рефакторинг на маленькие задачи
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект есть GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [x] Модули с GCD обмазываем @unchecked Sendable
- [x] Нарезаем их рефакторинг на маленькие задачи
- [ ] Создаём задачи по изоляции сложных модулей
- [ ] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## если у вас многомодульный проект есть GCD

- [x] Есть способы per module включение флага
- [x] Включаем флаг по умолчанию, на старых модулях выключаем
- [x] Поддерживаем флаг в простых модулях
- [x] Модули с GCD обмазываем @unchecked Sendable
- [x] Нарезаем их рефакторинг на маленькие задачи
- [x] Создаём задачи по изоляции сложных модулей
- [x] Раздаём задачи по сложным модулям желающим (и не желающим)

# Внедрение

## Если у вас монолит (Простой вариант)

- [ ] Смотрим доклад про Modularity Explorer
- [ ] Разбиваем проект на модули
- [ ] Теперь у вас многомодульный проект, возвращаемся к предыдущему чек-листу

От модуляризации к Clang и обратно

# Внедрение

## Если у вас монолит (Простой вариант)

- [ ] Смотрим доклад про Modularity Explorer
- [ ] Разбиваем проект на модули
- [ ] Теперь у вас многомодульный проект, возвращаемся к предыдущему чек-листу

# Внедрение

## Если у вас монолит (Простой вариант)

- [x] Смотрим доклад про Modularity Explorer
- [ ] Разбиваем проект на модули
- [ ] Теперь у вас многомодульный проект, возвращаемся к предыдущему чек-листу

# Внедрение

## Если у вас монолит (Простой вариант)

- [x] Смотрим доклад про Modularity Explorer
- [x] Разбиваем проект на модули
- [ ] Теперь у вас многомодульный проект, возвращаемся к предыдущему чек-листу

# Внедрение

## Если у вас монолит (Простой вариант)

- [x] Смотрим доклад про Modularity Explorer
- [x] Разбиваем проект на модули
- [x] Теперь у вас многомодульный проект, возвращаемся к предыдущему чек-листу

# Внедрение

## Если у вас монолит (Сложный вариант)

- [ ] Находим жертву добровольца
- [ ] Гига-заход на изоляцию, стараемся не ловить конфликты
- [ ] В сложных местах выставляем @unchecked Sendalbe
- [ ] Строим AST
- [ ] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Внедрение

## Если у вас монолит (Сложный вариант)

- [ ] Находим жертву добровольца
- [ ] Гига-заход на изоляцию, стараемся не ловить конфликты
- [ ] В сложных местах выставляем @unchecked Sendalbe
- [ ] Строим AST
- [ ] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Внедрение

## Если у вас монолит (Сложный вариант)

- [x] Находим жертву добровольца
- [ ] Гига-заход на изоляцию, стараемся не ловить конфликты
- [ ] В сложных местах выставляем @unchecked Sendalbe
- [ ] Строим AST
- [ ] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Внедрение

## Если у вас монолит (Сложный вариант)

- [x] Находим жертву добровольца
- [x] Гига-заход на изоляцию, стараемся не ловить конфликты
- [ ] В сложных местах выставляем @unchecked Sendalbe
- [ ] Строим AST
- [ ] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Внедрение

## Если у вас монолит (Сложный вариант)

- [x] Находим жертву добровольца
- [x] Гига-заход на изоляцию, стараемся не ловить конфликты
- [x] В сложных местах выставляем @unchecked Sendalbe
- [ ] Строим AST
- [ ] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Внедрение

## Если у вас монолит (Сложный вариант)

- [x] Находим жертву добровольца
- [x] Гига-заход на изоляцию, стараемся не ловить конфликты
- [x] В сложных местах выставляем @unchecked Sendalbe
- [x] Строим AST
- [ ] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Внедрение

## Если у вас монолит (Сложный вариант)

- [x] Находим жертву добровольца
- [x] Гига-заход на изоляцию, стараемся не ловить конфликты
- [x] В сложных местах выставляем @unchecked Sendalbe
- [x] Строим AST
- [x] Идём bottom-up по AST и заменяем @unchecked Sendable на Sendable/Акторы

# Выводы ?

# Выводы ?

🚀 Работать с адаптированым к Complete Concurrency Checking - Просто

🪜 Адаптировать проект можно итеративно

❌ Нужен структурный подход

🚋 Важно шарить знания на команду

# Выводы ?

🚀 Работать с адаптированым к Complete Concurrency Checking - Просто

🪜 Адаптировать проект можно итеративно

❌ Нужен структурный подход

🚃 Важно шарить знания на команду

# Выводы ?

🚀 Работать с адаптированым к Complete Concurrency Checking - Просто

🪜 Адаптировать проект можно итеративно

❌ Нужен структурный подход

🚆 Важно шарить знания на команду

# Выводы ?

🚀 Работать с адаптированым к Complete Concurrency Checking - Просто

🪜 Адаптировать проект можно итеративно

❌ Нужен структурный подход

🚃 Важно шарить знания на команду