

기술 공유 세션

장지현

2025.09.11

목차

1. Network Error

2. 앨런(LLM) 삽질 공유기

3. UI 순차 업데이트

4. 작업 취소

5. URL 링크 브라우저

Network Error

Core/Remote/NetworkError.swift

네트워크 에러를 세분화하게 된 이유

기존 네트워크 에러 case

invalidURL URL(string:) 실패 시

websocketError 웹소켓

invalidAPIKey xcconfig에서 키를 불러오지 못한 경우

invalidResponse 200번대가 아닌 모든 상태 코드

Network Error

Core/Remote/NetworkError.swift

네트워크 에러를 세분화하게 된 이유

Alan API의 상태 코드가 다양함 (401, 404, 414, 500)

기존에는 200번대 이외에는 `invalidResponse`로 처리

원인 파악을 위해 포스트맨 등의 API 테스트 툴을 거쳐야했음

외부 도구 없이 디버깅 불가능 !!

Network Error

Core/Remote/NetworkError.swift

```
1 /// 네트워크 에러를 정의한 객체입니다.
2 enum NetworkError: Error {
3     case networkError(_ error: URLError) // 일반 네트워크 오류
4     case taskCancelled // 작업 취소
5     case invalidURL // 잘못된 URL
6     case invalidResponse // 유효하지 않은 응답
7     case encodingError // 인코딩 오류
8     case decodingError(_ error: DecodingError) // 디코딩 오류
9     case invalidAPIKey // API 키 오류
10    case quotaExceeded(_ statusCode: Int) // 호출 한도 초과
11    case notFound(_ statusCode: Int) // 리소스 없음
12    case uriTooLong(_ statusCode: Int) // URI 길이 초과
13    case serviceUnavilable(_ statusCode: Int) // 서버 점검/장애
14    case serverError(_ statusCode: Int) // 서버 내부 오류
15    case remoteError(_ statusCode: Int, _ errorData: String) // 서버 전달 오류
16    case unknown(_ statusCode: Int) // 알 수 없는 상태 코드
17    case websocketError // 웹소켓 오류
18 }
```


연관값 (Associated Value)

열거형에서 각 case에 추가적인 데이터를 붙일 수 있게 해주는 값

```
1 enum MediaType {  
2     case image  
3     case video  
4     case text  
5 }
```

case마다 필요한 데이터를 다르게 가질 수 있음

```
1 enum Media {  
2     case image(fileName: String, resolution: (width: Int, height: Int))  
3     case video(fileName: String, duration: Double)  
4     case text(content: String)  
5 }
```

원시값 (Raw Value)

enum 전체에서 동일한 타입의 기본값을 가짐

```
1 enum Direction: String {  
2     case north = "북"  
3     case south = "남"  
4     case east  = "동"  
5     case west  = "서"  
6 }  
7  
8 let dir = Direction.north  
9 print(dir.rawValue) // "북"
```

연관값 (Associated Value)

case별로 다른 타입, 다른 개수의 값을 가질 수 있음

case를 사용할 때마다 새로운 값을 넣을 수 있음

```
1 enum NetworkResult {  
2     case success(data: String)  
3     case failure(errorCode: Int, message: String)  
4 }  
5  
6 let result1 = NetworkResult.success(data: "성공 데이터")  
7 let result2 = NetworkResult.failure(errorCode: 404, message: "Not Found")
```


Network Error

Core/Remote/NetworkError.swift

NetworkError.swift

```
1 extension NetworkError: LocalizedError {
2     /// 사용자에게 표시할 오류 설명입니다.
3     ///
4     /// - `taskCancelled`: "작업이 취소되었습니다. 아래 버튼을 눌러 다시 시도해 주세요."
5     /// - 그 외: "데이터를 불러오는 데 실패했어요. 잠시 후 다시 시도해 주세요."
6     var errorDescription: String? {
7         switch self {
8             case .taskCancelled:
9                 return "작업이 취소됐어요"
10            default:
11                return "데이터를 불러오지 못했어요\n잠시 후 다시 시도해 주세요"
12        }
13    }
14 }
```

LocalizedError protocol

```
public protocol LocalizedError : Error {
    // 오류에 대한 간단한 설명
    var errorDescription: String? { get }

    // 오류가 발생한 이유
    var failureReason: String? { get }

    // 사용자가 시도할 수 있는 해결 방법
    var recoverySuggestion: String? { get }

    // 추가 문서나 도움말 anchor
    var helpAnchor: String? { get }
}
```

error.localizedDescription

Network Error

Core/Remote/NetworkError.swift

```
1 extension NetworkError {
2     /// 오류 정보를 문자열로 변환하여 로그 형식으로 반환합니다.
3     ///
4     /// - Parameters:
5     ///   - file: 호출한 파일 경로 (`#fileID`로 기본값 지정)
6     ///   - function: 호출한 함수 이름 (`#function`으로 기본값 지정)
7     /// - Returns: 오류 유형과 호출 위치, `.decodingError`의 경우 디버그 설명을 포함한 문자열
8     func log(file: String = #fileID, function: String = #function) -> String {
9         /// DecodingError인 경우 debugDescription을 log로 출력합니다.
10        if case .decodingError(let err) = self {
11            let message: String
12            switch err {
13            case .typeMismatch(_, let context),
14                 .valueNotFound(_, let context),
15                 .keyNotFound(_, let context),
16                 .dataCorrupted(let context):
17                message = context.debugDescription
18            @unknown default:
19                message = "알 수 없는 디코딩 오류입니다"
20            }
21            let escaped = message.replacingOccurrences(of: "\"", with: "\\\"")
22            return "decodingError(\"\"(escaped)\")-\"\"(file)#\"\"(function)\""
23        }
24
25        return "\"(String(describing: self))-\"\"(file)#\"\"(function)\""
26    }
27 }
```

```
print(error.log())
```

1. .decodingError가 아니라면

NetworkError case 이름 `String(describing: self)`

파일 (기본값 `#fileID`) `file`

함수 (기본값 `#function`) `function`

2. .decodingError라면

err를 통해 DecodingError(Swift 표준)를 꺼내옴

4가지 케이스가 있고, 각 케이스는 DecodingError.Context를 가짐

Context 구조체 - debugDescription을 가져옴

가져온 message에서 " 를 \" 로 변경해 출력이 깨지는 걸 방지

Network Error

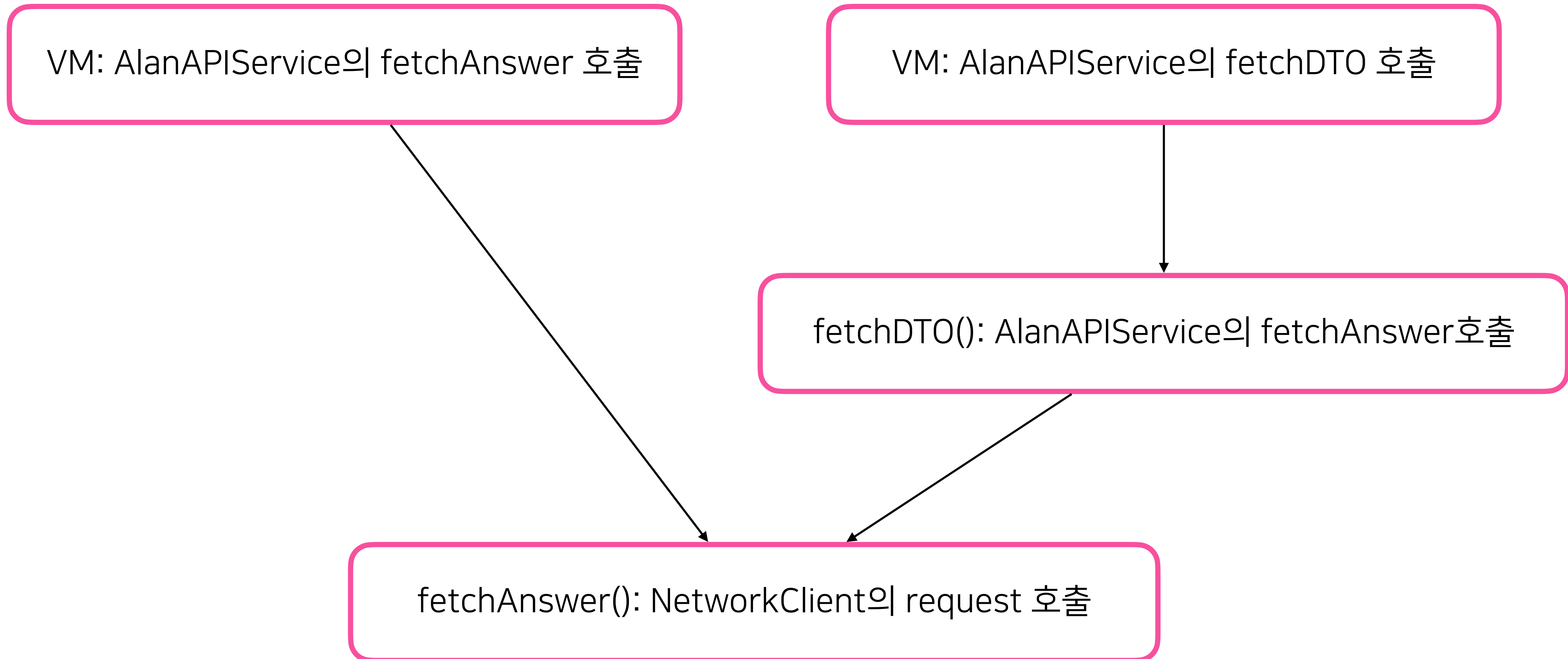
Core/Remote/NetworkError.swift

`decodingError("Expected String but found Int instead")-"NetworkClient.swift#request(url:)"`

`notFound(404)-"Project/MyService.swift#fetchData"`

`taskCancelled-"Project/MyService.swift#fetchData"`

Network Error



Network Error

Data/API/Alan/AlanAPIService.swift

```
func fetchAnswer(content: String, action: AlanAction) async throws -> AlanResponseDTO {  
    guard let clientID = switchClientID(for: action), !clientID.isEmpty else { throw NetworkError.invalidAPIKey }  
  
    let urlString = "\(endpoint)?content=\(content)&client_id=\(clientID)"  
    guard let url = URL(string: urlString) else { throw NetworkError.invalidURL }  
    let alanResponseDTO: AlanResponseDTO = try await network.request(url: url)  
  
    return alanResponseDTO  
}
```

```
private func fetchDTO<T: Decodable>(prompt: Prompt, action: AlanAction) async throws -> T {  
    let answer = try await fetchAnswer(content: prompt.content, action: action)  
  
    guard let jsonData = answer.content.extractedJSON.data(using: .utf8) else {  
        throw NetworkError.encodingError  
    }  
  
    do {  
        return try JSONDecoder().decode(T.self, from: jsonData)  
    } catch let decodingError as DecodingError {  
        throw NetworkError.decodingError(decodingError)  
    }  
}
```

Network Error

Core/Remote/HTTP/NetworkClient.swift

```
final class NetworkClient {  
    func request<T: Decodable>(url: URL) async throws -> T { ... }  
  
    private func handleStatusCode(_ statusCode: Int, data: Data) throws { ... }  
}
```

```
func request<T: Decodable>(url: URL) async throws -> T {  
    do {  
        let (data, response) = try await URLSession.shared.data(from: url)  
  
        guard let httpResponse = response as? HTTPURLResponse else {  
            throw NetworkError.invalidResponse  
        }  
        let statusCode = httpResponse.statusCode  
        try handleStatusCode(statusCode, data: data)  
  
        do {  
            return try JSONDecoder().decode(T.self, from: data)  
        } catch let decodingError as DecodingError {  
            throw NetworkError.decodingError(decodingError)  
        }  
    } catch let urlError as URLError where urlError.code == .cancelled {  
        throw NetworkError.taskCancelled  
    } catch let urlError as URLError {  
        throw NetworkError.networkError(urlError)  
    } catch {  
        throw error  
    }  
}
```

Network Error

Core/Remote/HTTP/NetworkClient.swift

```
final class NetworkClient {  
    func request<T: Decodable>(url: URL) async throws -> T { ... }  
  
    private func handleStatusCode(_ statusCode: Int, data: Data) throws { ... }  
}
```

```
func request<T: Decodable>(url: URL) async throws -> T {  
    do {  
        let (data, response) = try await URLSession.shared.data(from: url)  
  
  
  
  
  
  
  
  
  
    } catch let urlError as NSError where urlError.code == .cancelled {  
        throw NetworkError.taskCancelled  
    } catch let urlError as NSError {  
        throw NetworkError.networkError(urlError)  
    } catch {  
        throw error  
    }  
}
```

네트워크 요청이 정상적으로 완료되지 않음

URLError

연결 관련 (인터넷 연결 없음, 호스트에 연결할 수 없음, 연결 끊어짐, ...)

요청 취소, 시간초과

보안/인증 관련 (SSL/TLS 보안 연결 실패, 서버 인증서 신뢰되지 않음, ...)

데이터 전송 관련 (리소스를 사용할 수 없음, 요청 데이터가 너무 큼, ...)

기타 (알 수 없음, 서버의 잘못된 응답, ...)

Network Error

Core/Remote/HTTP/NetworkClient.swift

```
final class NetworkClient {  
    func request<T: Decodable>(url: URL) async throws -> T { ... }  
  
    private func handleStatusCode(_ statusCode: Int, data: Data) throws { ... }  
}
```

```
let (data, response) = try await URLSession.shared.data(from: url)
```

```
guard let httpResponse = response as? HTTPURLResponse else {  
    throw NetworkError.invalidResponse  
}
```

response를

HTTPURLResponse로 캐스팅할 수 없음

Network Error

Core/Remote/HTTP/NetworkClient.swift

```
final class NetworkClient {  
    func request<T: Decodable>(url: URL) async throws -> T { ... }  
  
    private func handleStatusCode(_ statusCode: Int, data: Data) throws { ... }  
}
```

```
let (data, response) = try await URLSession.shared.data(from: url)
```

```
guard let httpResponse = response as? HTTPURLResponse else {  
    throw NetworkError.invalidResponse  
}
```

```
let statusCode = httpResponse.statusCode  
try handleStatusCode(statusCode, data: data)
```

handleStatusCode

Network Error

Core/Remote/HTTP/NetworkClient.swift

```
private func handleStatusCode(_ statusCode: Int, data: Data) throws {
    switch statusCode {
    case 200..<300:
        return
    case 401:
        throw NetworkError.quotaExceeded(statusCode)
    case 404:
        throw NetworkError.notFound(statusCode)
    case 414:
        throw NetworkError.uriTooLong(statusCode)
    case 503:
        throw NetworkError.serviceUnavilable(statusCode)
    case 500..<600:
        throw NetworkError.serverError(statusCode)
    default:
        if let errorData = String(data: data, encoding: .utf8) {
            throw NetworkError.remoteError(statusCode, errorData)
        } else {
            throw NetworkError.unknown(statusCode)
        }
    }
}
```

Network Error

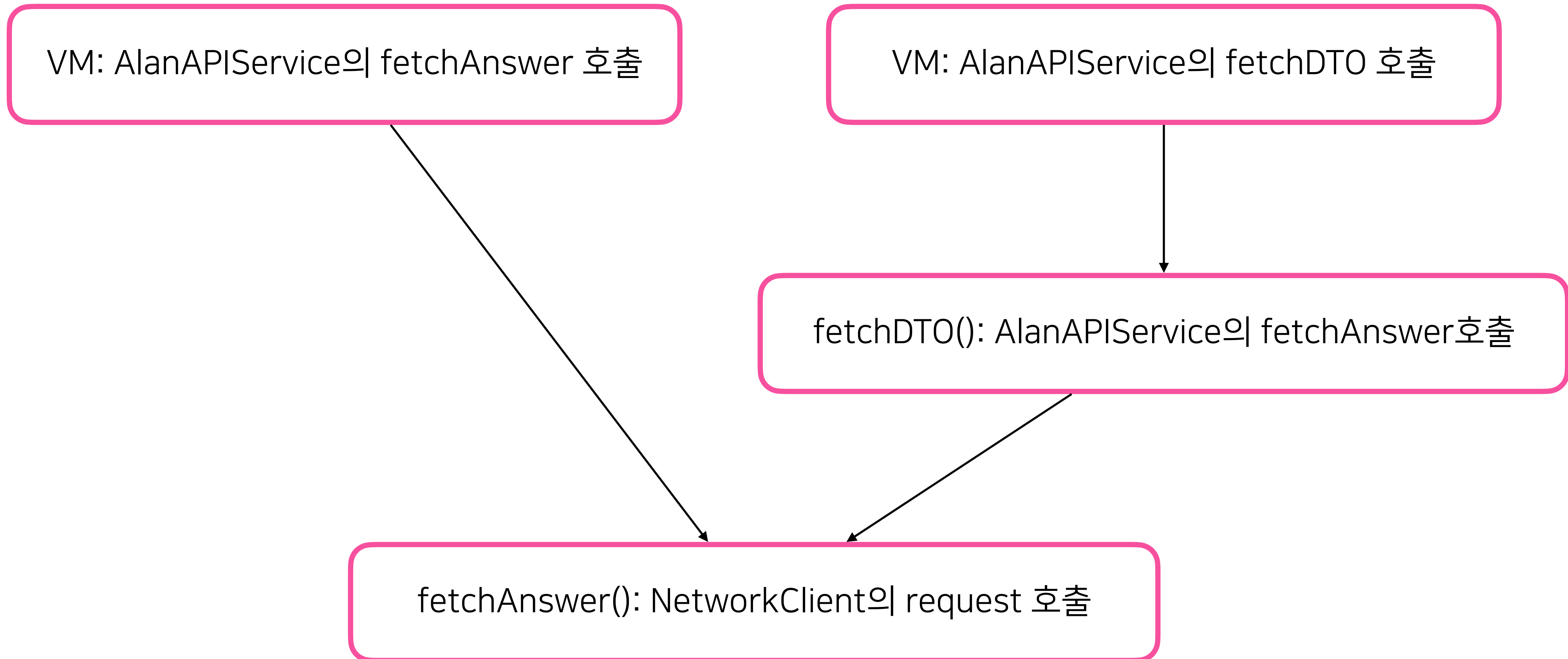
Core/Remote/HTTP/NetworkClient.swift

```
final class NetworkClient {  
    func request<T: Decodable>(url: URL) async throws -> T { ... }  
  
    private func handleStatusCode(_ statusCode: Int, data: Data) throws { ... }  
}
```

응답 decoding 실패

```
do {  
    return try JSONDecoder().decode(T.self, from: data)  
} catch let decodingError as DecodingError {  
    throw NetworkError.decodingError(decodingError)  
}
```

Network Error



앨런 삽질 공유기

```
{  
  "action": {  
    "name": "search_web",  
    "speak": "검색 결과를 바탕으로 답변을 생성하고 있어요."  
  },  
  "content": "String Result"  
}
```

1. 얻고싶은 데이터 생각
2. UI 표현 방식 생각
3. DTO를 만들어서 앨런에게 공유

앨런 삽질 공유기

비트코인에 대한 심볼, 웹사이트, 최초 발행일, 소개를 JSON 형식으로 답변해줘~

```
struct CoinOverview {  
    let symbol: String  
    let websiteURL: String?  
    let launchDate: String  
    let description: String  
}
```

앨런 삽질 공유기

```
struct CoinOverview { // 주석은 주어진 키워드가 '이더리움'인 경우 예상 응답
    /// 심볼: ETH
    let symbol: String

    /// 웹사이트: https://ethereum.org/ko/
    let websiteURL: String?

    /// 최초발행: 2015.07.
    let startDate: String

    /// 디지털 자산 소개, 문장을 자연스럽게 연결하기 :
    /// 이더리움은 비탈릭 부테린이 개발한 블록체인 기반의 분산 컴퓨팅 플랫폼이자 운영 체제입니
다. 스마트 계약 기능을 제공하여 다양한 탈중앙화 애플리케이션(DApps)을 개발할 수 있도록 지원합니
다. 이더리움의 네이티브 암호화폐인 이더(ETH)는 네트워크에서의 거래 수수료 및 계산 서비스를 위한 연
료로 사용됩니다. 이더리움은 블록체인 기술의 활용 범위를 확장하여 디지털 자산의 새로운 가능성을 열었
습니다.
    let description: String
}
```

1. 답변할 항목은 위에 제공한 Swift 구조체의 주석에 따라 내용을 구성
2. JSON 형식과 변수 또한 구조체와 통일
3. 마크다운 문법 사용 금지
4. 위 세가지 규칙을 적용해서 코인 ""에 대한 개요를 JSON 형식으로 답변해줘.

앨런 삽질 공유기

```
struct CoinOverviewDTO: Codable {  
    let symbol: String  
    let websiteURL: String?  
    let launchDate: String  
    let description: String  
}
```

"\((coinKName))" 개요를 위 JSON 형식으로 작성 (마크다운 금지, 출처 제외)

앨런 삽질 공유기

```
struct CoinTodayNewsDTO: Codable {  
    let summaryOfTodaysMarketSentiment: String  
    let articles: [CoinArticleDTO] // 3개  
}
```

오늘 시장 분위기 살펴보기

비트코인은 최근 아마존을 제치고 세계 5위 자산으로 올라서는 성과를 달성했으나, 최근 5일 연속 가격이 하락하며 3주 만에 최저치를 기록했습니다. 8월은 비수기로 예상되며, 가격이 11만 2000달러까지 하락할 가능성이 제기되고 있습니다. JP모건 CEO는 비트코인에 대한 회의적인 입장을 밝히면서도 고객의 선택을 존중하겠다는 입장을 표명했습니다. 가상화폐 시장 전반이 하락세를 보이고 있으며, 주요 코인들이 큰 폭으로 하락하고 있습니다. 시장 분위기는 전반적으로 부정적이며, 가격 하락과 비수기 전망으로 인해 투자자들의 불안감이 커지고 있습니다.

오늘 시장 분위기 살펴보기

긍정적

1. 현재 국내 시간을 기준으로 최근 24시간 뉴스를 분석해 \((coinKName)시장 분위기를 요약

위 조건에 따라 "\((coinKName))"에 대한 내용을 위 JSON 형식으로 작성 (마크다운 금지, 답변에 출처 금지)

```
struct CoinWeeklyDTO: Codable {  
    let priceTrend: String  
    let volumeChange: String  
    let reason: String  
}
```

주간 동향 확인

- 가격 추이: 비트코인의 가격은 일주일 동안 약 2.8% 하락하였으며, 113,000달러대에서 변동을 보였습니다.
- 거래량 변화: 최근 일주일 동안 비트코인의 거래량은 변동성이 있었으며, 특히 가격 하락 시기에 '저가 매수' 활동이 증가하면서 거래량이 일시적으로 증가하는 경향을 보였습니다.

주간 동향 확인

- 가격 추이: 상승세
- 거래량 변화: 증가

1. 현재 국내 시간을 기준으로 일주일 동안의 정보 사용

위 조건에 따라 "\((coinKName))"에 대한 내용을 위 JSON 형식으로 작성 (마크다운 금지, 출처 제외)

앨런 삽질 공유기

현재 "비트코인"에 대한 최신 뉴스를 가져오는 데 어려움이 있습니다. 대신 비트코인 시장의 일반적인 경향이나 다른 정보를 제공해드릴 수 있으니, 다른 질문이 있다면 말씀해 주세요.
오류 발생: The data couldn't be read because it isn't in the correct format.

```
HTTP Alan / Coin Weekly

GET {{baseUrl}} ?content=struct CoinWeeklyDTO: Codable { /// 최근 일주일 가격 추이 let priceTrend: String /// 최근 일주일 거래량 변화 let volumeChange: String /// 지난 일주일간 가격 추이와 거래량 변화의 주요 원인 let reason: String)

{
  "content": "{\n  \"summaryOfTodaysMarketSentiment\": \"최근 비트코인을 제치고 세계 5위 자산으로 올라서는 성과를 달성했으나, 최근 5일 연속 가격이 하락하며 3주 만에 최저치를 기록했습니다. 8월은 비수기인 만큼 비트코인에 대한 회의적인 입장을 밝히면서도 고객의 선택을 존중하겠습니다. 가상화폐 시장 전반이 하락세를 보이고 있으며, 주요 코인들이 큰 폭으로 하락하고 있습니다. 시장 분위기는 전반적으로 부정적이며, 가격 변동성이 커졌습니다.\"\n  {\n    \"title\": \"비트코인 다시 12만달러 넘길까...상승 재개 신호 나왔다\", \n    \"summary\": \"비트코인이 최근 하락세를 보였지만, 상승 재개 신호가 포착되었습니다.\"\n  }, \n  {\n    \"title\": \"이렇게 비싼데 또?...스트래티지, 비트코인 2.1만개 매입\", \n    \"summary\": \"스트래티지가 비트코인 2.1만개를 추가 매입하며 시장의 주목을 받고 있습니다.\"\n  }, \n  {\n    \"title\": \"저주...비트코인 폭락 가능성 언급\", \n    \"summary\": \"로버트 기요사키가 비트코인 8월 폭락 가능성을 경고하며 시장의 관심을 끌고 있습니다.\"\n  }\n}"
```


앨런 삽질 공유기

위 다섯가지 규칙을 적용해서 "이더리움"에 대한 오늘 시장 분위기 요약본과 뉴스 배열을 제공해줘.

```
:Sentiment\": \ '엑스알피(XRP)는 ????? 4시간 동안 가격이 소폭 상승하며  
전한 상황입니다.\", \n \"articles\": [\n    {\n        \"title\
```

앨런 삽질 공유기

```
1 {
2   "action": {
3     "name": "",
4     "speak": "질문의 의도를 이해하고 있어요.",
5   },
6   "content": "죄송합니다. 뉴스 원문 링크를 제공할 수 있도록 하겠습니다. 다음은 오늘자 이더리움에 대한 중요한 뉴스 3개의 헤드라인과 내용 요약, 원문 링크입니다:\n\n1. \n   - **title**:\n   \"이더리움에 쏠리는 월가,\n   집중적인 매수로 인해 강세를 보이고 있습니다.\n   - **url**:\n   [https://example-news-site.com/article/374](https://example-news-site.com/article/374)\n\n2. \n   - **title**:\n   \"\n   이더리움 ETF에서 하루 만에 5200억원이 빠져나가며 사상 최대 순유출을 기록했습니다.\n   - **url**:\n   [https://example-news-site.com/article/373](https://example-news-site.com/article/373)\n\n3. \n   - **title**:\n   \"\n   이더리움 ETF가 3400달러 부근에서 지지를 확보한 뒤 반등세를 보이며 3800달러 돌파 가능성을 높이고 있습니다.\n   - **url**:\n   [https://example-news-site.com/article/376](https://example-news-site.com/article/376)\n   사이트에서 확인하실 수 있습니다.\"
7 }
```

```
"action": {
  "name": "",
  "speak": "질문의 의도를 이해하고 있어요.",
},
"content": "네, 맞습니다. 현재로서는 실제 뉴스 원문 링크를 제공할 수 없습니다. 뉴스 요약과 헤드라인을 기반으로 정보를 제공할 수 있지만, 링크는 직접 확인하셔야 합니다. 불편을 드려 죄송합니다."
```


앨런 샵질 공유기

```
{
  "summaryOfTodaysMarketSentiment": "비트코인은 미국 소비자물가지수(CPI) 발표를 앞두고 신중한 분위기를 보이고 있습
  벌 경제 불확실성과 금리 정책에 대한 우려가 여전히 존재하여 투자자들의 주의가 필요합니다.",

  "articles": [

    {

      "title": "미국 물가지수 발표 앞두고 비트코인 시장 분위기 '신중'",

      "summary": "미국의 물가지수 발표를 앞두고 비트코인 시장이 신중한 분위기를 보이고 있습니다.",

      "newsSourceURL": "https://x.com/LilMoonLambo/status/1952056698904150282"

    },

    {

      "title": "비트코인 시장, 미국 CPI 앞두고 저항선 돌파 시도 분석",

      "summary": "비트코인 시장 내 12만 2천 달러 돌파 시도가 지속되고 있다는 분석이 나왔습니다.",

      "newsSourceURL": "https://x.com/LilMoonLambo/status/1952056698904150282"

    },

    {

      "title": "8월은 비트코인 하락의 달? 역사적 하락 통계 vs 현재 강세 분위기, 어디에 베팅할까",

      "summary": "비트코인이 사상 최고가를 경신하며 강세를 보이고 있으나, 8월은 역사적으로 하락세를 보이는 달입니다.",

      "newsSourceURL": "https://x.com/LilMoonLambo/status/1952056698904150282"

    }

  ]
}
```

LilMoonLambo

@LilMoonLambo

...

This is the most accurate crypto meme of the week

[게시물 번역하기](#)

China BANS Bitcoin

2017

2018

2019

2020

2021

2022

2023

2024

2025

오전 2:19 · 2025년 8월 4일 · 1.1만 조회수

35

10

108

3

로는 강세를 유지하고 있습니다. 그러나 글로

앨런 삽질 공유기

그치만 출처를 빼달라고 했어도 출처를 붙여주는 경우는 있지~

Coin Today and News 교육

Save Share GET Coin OverView 교육

{{baseUrl}} ?content=방금 한 답변의 출처 부탁해&client_id=7f65bb01-3a70-42c9-91d5-bad4c1fc9310

Send

Authorization Headers (6) Body Scripts Settings

	Value	Description	Bulk Edit
content	방금 한 답변의 출처 부탁해		
client_id	4095edf0-0fde-405e-8463-736a9d06f...		
client_id	7f65bb01-3a70-42c9-91d5-bad4c1fc9310		
	Value	Description	

Coin Today and News 교육

Save Share GET Coin OverView 교육

{{baseUrl}} ?content=근데 답변을 구성하는데 사용된

Send

Authorization Headers (6) Body Scripts Settings

	Value	Description	Bulk Edit
content	근데 답변을 구성하는데 사용된 출처가 있잖아 그...		
client_id	4095edf0-0fde-405e-8463-736a9d06f...		
client_id	7f65bb01-3a70-42c9-91d5-bad4c1fc9310		
	Value	Description	

새로운 소식들이 있어요

전체적인 시장은 **호재**예요

암호화폐 시장 시가총액이 4조 달러를 돌파하며 상승세를 이어가고 있습니다. [\[\(출처133\)\]\(https://kr.beincrypto.com/base-news/124941/\)](https://kr.beincrypto.com/base-news/124941/)

이더리움이 4,300달러를 돌파하며 2021년 말 이후 최고치를 경신했습니다. [\[\(출처137\)\]\(https://www.panewslab.com/ko/articles/4431e392-7b32-415d-a4b7-28e1be388e5d\)](https://www.panewslab.com/ko/articles/4431e392-7b32-415d-a4b7-28e1be388e5d)
애플의 투자 발표로 인해 미국 증시와 암호화폐 시장이 동반 반등했습니다. [\[\(출처138\)\]\(http://www.theonetcv.kr/news/articleView.html?idxno=243820\)](http://www.theonetcv.kr/news/articleView.html?idxno=243820)

지금 주요 커뮤니티 분위기는

현재 시스템에서는 출처를 직접 제공할 수 있는 기능이 제한

출처에 대한 접근이 불가능

Headers (6) Test Results

Preview Visualize

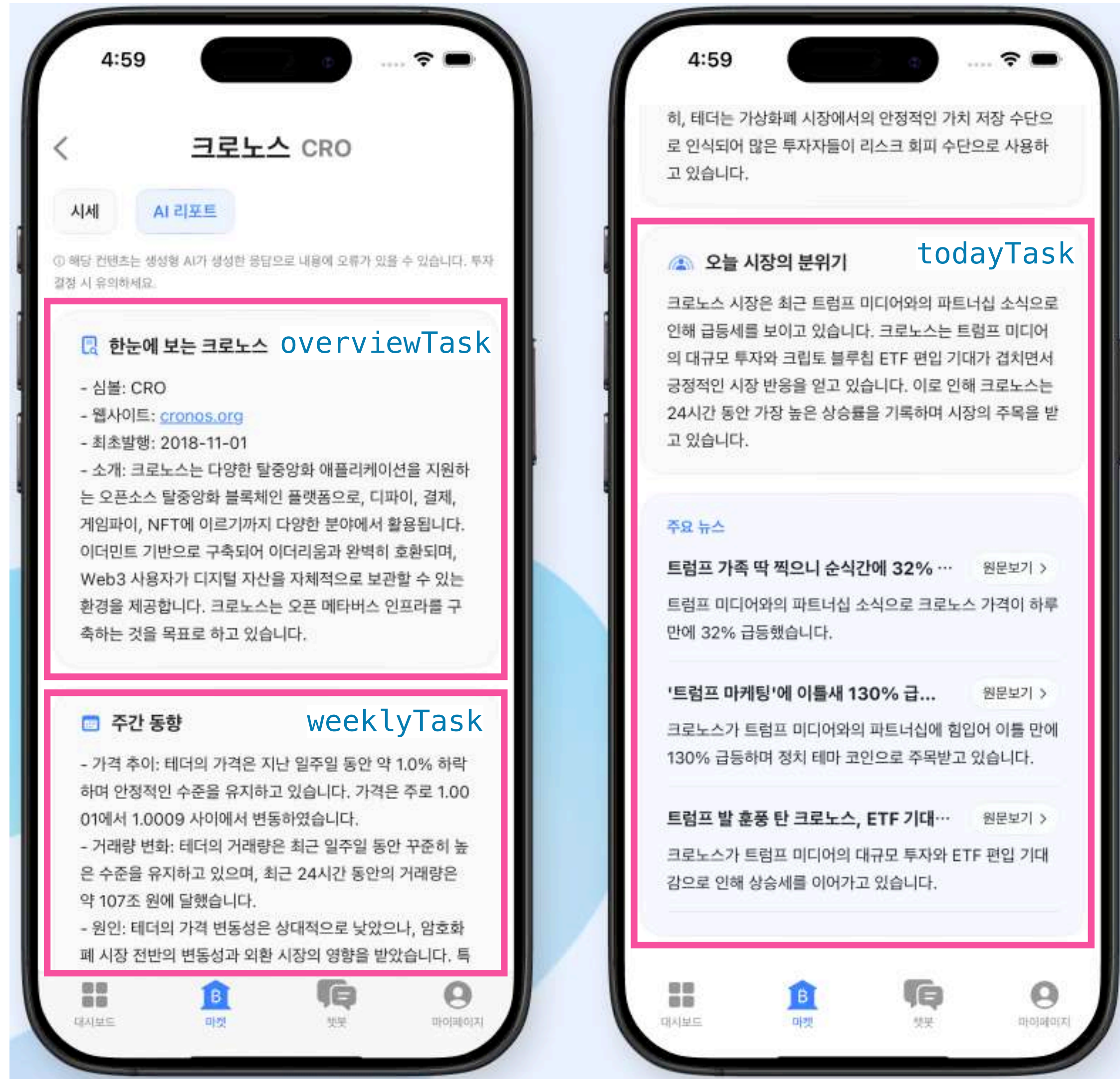
```
"action": {
  "name": "summarize_references",
  "speak": "검색 결과 중 요청하신 출처를 더 자세히 읽고 있어요."
},
"content": "현재 시스템에서는 출처를 직접 제공할 수 있는 기능이 제한되어 있어, 답변에 사용된 출처를 제공할 수 없습니다. 이 점 양해 부탁드립니다. 추가적인 정보나 도움이 필요하시면 언제든지 말씀해 주세요."
```

Headers (6) Test Results

Preview Visualize

```
"action": {
  "name": "summarize_references",
  "speak": "검색 결과 중 요청하신 출처를 더 자세히 읽고 있어요."
},
"content": "현재 시스템에서는 답변을 구성하는 데 사용된 출처를 직접 전달할 수 있는 기능이 제한되어 있습니다. 출처에 대한 접근이 불가능하여 링크를 제공할 수 없는 점 양해 부탁드립니다. 답변에 필요한 정보를 최대한 정확하게 제공하기 위해 노력하고 있습니다. 추가적인 도움이 필요하시면 언제든지 말씀해 주세요."
```


UI 순차 업데이트



한눈에 보는 ~~

주간 동향

오늘 시장의 분위기 + 주요뉴스

UI 순차 업데이트

1차

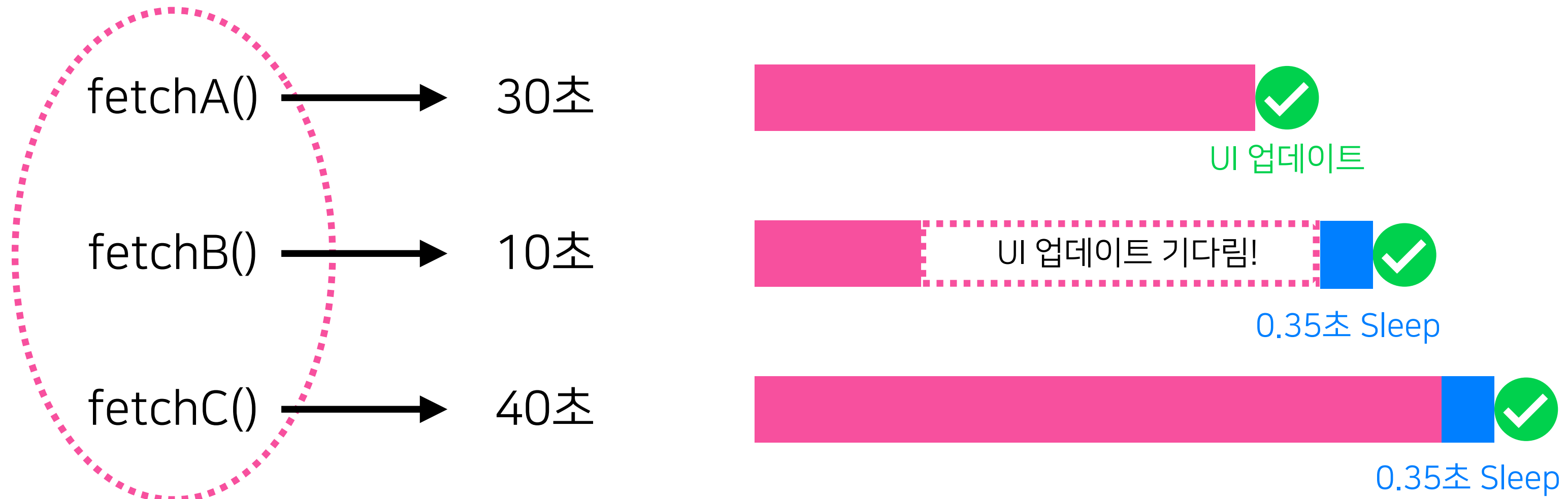
```
init {  
    fetchA()  
    fetchB()  
    fetchC()  
}  
  
fetchA {  
    Task {  
        네트워크 요청  
        UI 업데이트  
    }  
}
```

2차

```
init {  
    Task {  
        await fetchA()  
        await fetchB()  
        await fetchC()  
    }  
}  
  
fetchA async {  
    네트워크 요청  
    UI 업데이트  
}
```

UI 순차 업데이트

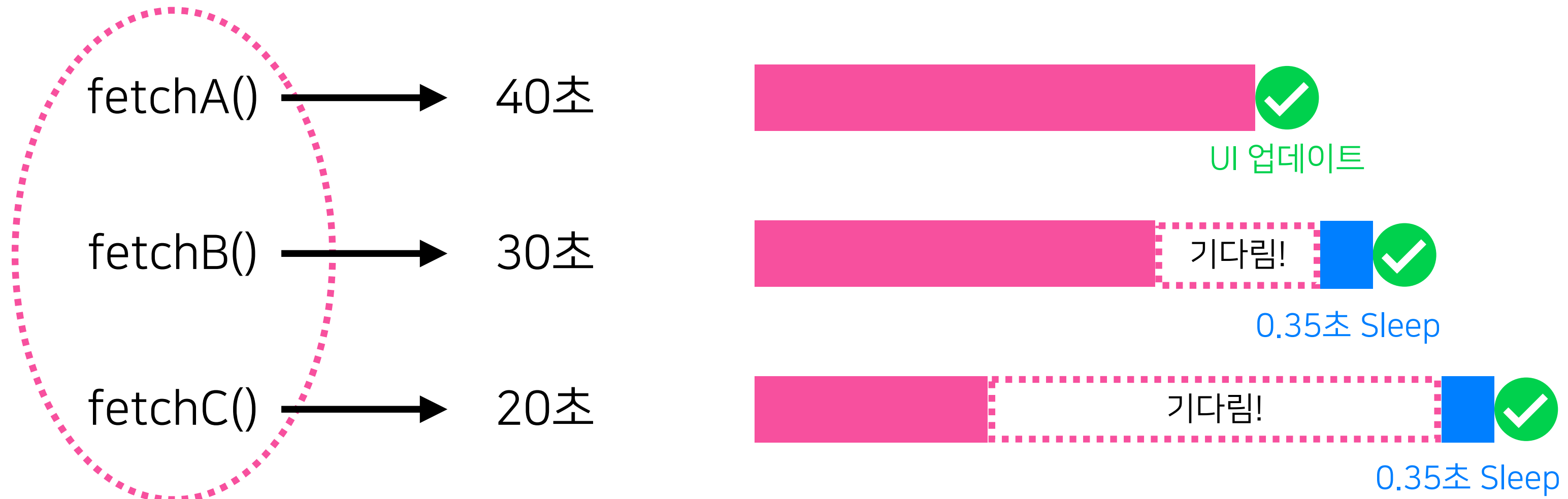
A → B → C



네트워크는 동시에!

UI 순차 업데이트

A → B → C



네트워크는 동시에!

UI 순차 업데이트

```
func load() async { .task()
```

```
    await MainActor.run {  
        overview = .loading  
        weekly = .loading  
        today = .loading  
    }
```

로딩으로 상태 변경

```
enum FetchState<Value> {  
    case loading  
    case success(Value)  
    case cancel(NetworkError)  
    case failure(NetworkError)  
}
```

```
overviewTask = Task { ... }
```

각 Task 병렬 실행 + 취소 감지

```
weeklyTask = Task { [weak self] in  
    try await withTaskCancellationHandler(  
        operation: {  
            guard let self else { throw CancellationError() }  
            return try await self.alanAPIService.fetchWeeklyTrends(for: self.coin)  
        },  
        onCancel: { [weak self] in  
            self?.weekly = .cancel(.taskCancelled)  
        },  
        isolation: MainActor.shared  
    )  
}
```

```
todayTask = Task { ... }
```

```
    ...  
}
```

UI 순차 업데이트

```
func load() async {  
    .task()  
  
    ...  
  
    await updateOverviewUI()  
    try? await Task.sleep(for: .milliseconds(350)) // UI가 순차적으로 적용되는 효과를 주기 위한 딜레이  
    await updateWeeklyUI()  
    try? await Task.sleep(for: .milliseconds(350)) // UI가 순차적으로 적용되는 효과를 주기 위한 딜레이  
    await updateTodayUI()  
}
```

UI 순차 업데이트

```
enum FetchState<Value> {  
    case loading  
    case success(Value)  
    case cancel(NetworkError)  
    case failure(NetworkError)  
}
```

```
private func updateOverviewUI() async {  
    await TaskResultHandler.apply(  
        of: overviewTask,  
        using: { data in  
            data.overview  
        },  
        update: { [weak self] state in  
            self?.overview = state  
        }  
    )  
}
```

UI 업데이트 코드

UI 순차 업데이트

```
static func apply<Success, Output>(
    of task: Task<Success, Error>?,    해당하는 task
    using transform: @Sendable (Success) async throws -> Output,    성공 결과를 출력 타입으로 변환
    update: @escaping (FetchState<Output>) -> Void,    UI 업데이트 코드
    sideEffect: ((Success) -> Void)? = nil    성공 시 추가 UI 업데이트 (선택)
) async {
    ...
}
```

UI 순차 업데이트

```
do {  
    let value = try await task?.value    UI에 해당하는 task의 결과가 나올 때까지 기다렸다가  
    if let value {  
        let output = try await transform(value)    파라미터로 받은 데이터를  
        await MainActor.run { update(.success(output)) }    받아서 UI 업데이트에 사용  
        if let sideEffect {  
            await MainActor.run { sideEffect(value) }  
        }    그 외 다른 데이터를 활용한 추가 UI 업데이트가 있다면 실행  
    }  
}  
} catch {  
    ...  
}
```

UI 순차 업데이트

```
do {
```

```
...
```

```
} catch {
```

```
if err
```

```
av
```

```
re
```

```
}
```

```
if let
```

```
pr
```

```
av
```

```
} else
```

```
pr
```

```
}
```

```
}
```



Error+Cancellation.swift

```
1 var isTaskCancellation: Bool {
2     if self is CancellationError { return true }
3
4     if let urlErr = self as? URLError, urlErr.code == .cancelled { return true }
5
6     if let ne = self as? NetworkError {
7         switch ne {
8             case .taskCancelled:
9                 return true
10            default:
11                return false
12        }
13    }
14    return false
15 }
```

UI 순차 업데이트

```
do {  
    ...  
} catch {  
    if error.isTaskCancellation { Task가 취소된 경우  
        await MainActor.run { update(.cancel(.taskCancelled)) }  
        return  
    }  
    if let ne = error as? NetworkError { NetworkError라면!  
        print(ne.log())  
        await MainActor.run { update(.failure(ne)) }  
    } else { 그 외 정의되지 않은 Error  
        print(error)  
    }  
}
```

UI 순차 업데이트

```
func load() async {
    cancelAll()

    await MainActor.run {
        overview = .loading
        weekly = .loading
        today = .loading
    }

    overviewTask = Task { try await alanAPIService.fetchOverview(for: coin) }

    weeklyTask = Task { [weak self] in
        try await withTaskCancellationHandler(
            operation: {
                guard let self else { throw CancellationError() }
                return try await self.alanAPIService.fetchWeeklyTrends(for: self.coin)
            },
            onCancel: { [weak self] in
                self?.weekly = .cancel(.taskCancelled)
            },
            isolation: MainActor.shared
        )
    }

    todayTask = Task { [weak self] in
        try await withTaskCancellationHandler(
            operation: {
                guard let self else { throw CancellationError() }
                return try await self.alanAPIService.fetchTodayNews(for: self.coin)
            },
            onCancel: { [weak self] in
                self?.today = .cancel(.taskCancelled)
            },
            isolation: MainActor.shared
        )
    }

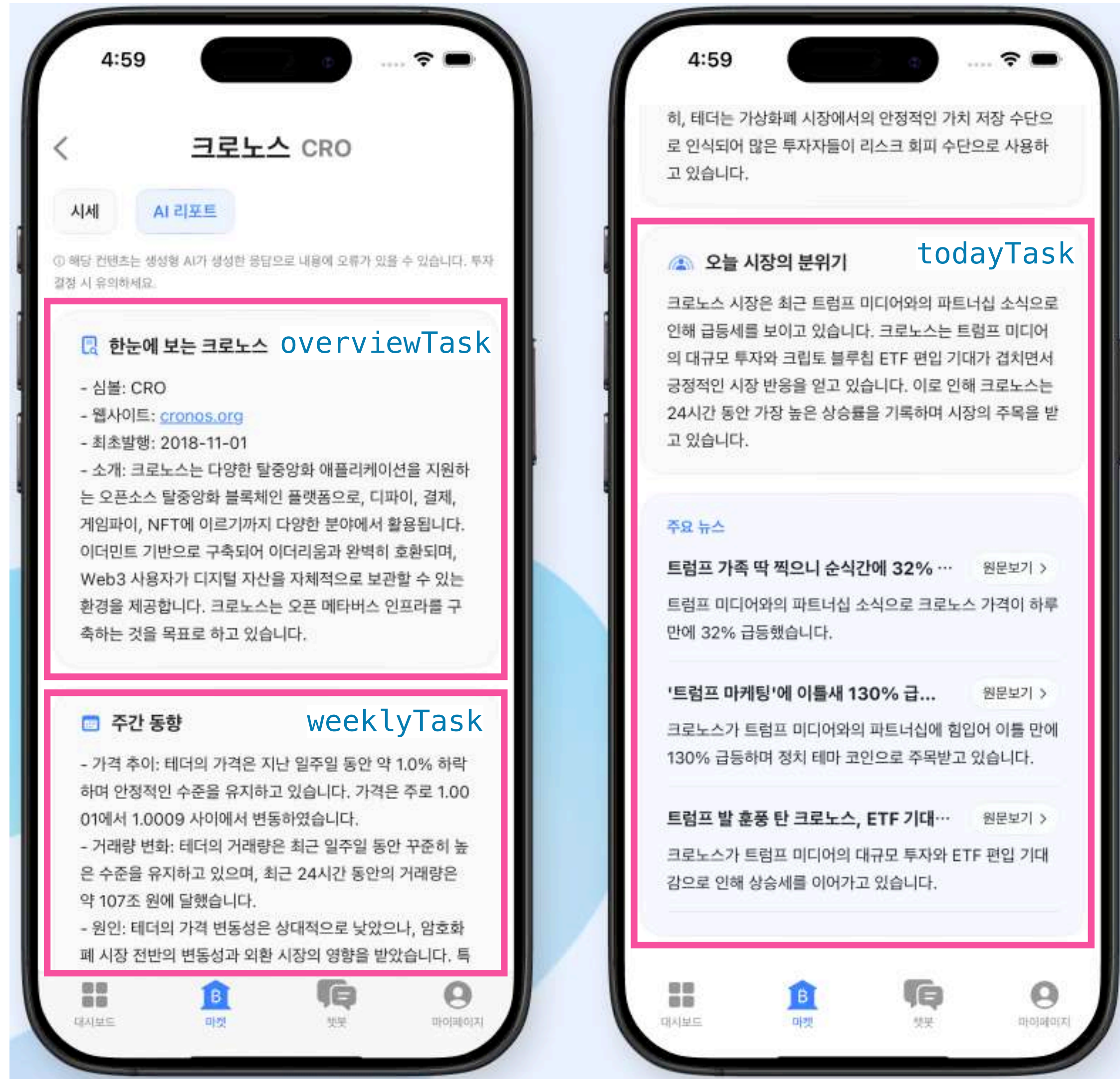
    await updateOverviewUI()
    try? await Task.sleep(for: .milliseconds(350)) // UI가 순차적으로 적용되는 효과를 주기 위한 딜레이
    await updateWeeklyUI()
    try? await Task.sleep(for: .milliseconds(350)) // UI가 순차적으로 적용되는 효과를 주기 위한 딜레이
    await updateTodayUI()
}
```

```
private func updateOverviewUI() async {
    await TaskResultHandler.apply(
        of: overviewTask,
        using: { data in
            data.overview
        },
        update: { [weak self] state in
            self?.overview = state
        }
    )
}

private func updateWeeklyUI() async {
    await TaskResultHandler.apply(
        of: weeklyTask,
        using: { data in
            data.weekly
        },
        update: { [weak self] state in
            self?.weekly = state
        }
    )
}

private func updateTodayUI() async {
    await TaskResultHandler.apply(
        of: todayTask,
        using: { data in
            data.today
        },
        update: { [weak self] state in
            self?.today = state
        },
        sideEffect: { [weak self] data in
            self?.news = data.articles.map { CoinArticle(from: $0) }
        }
    )
}
```


작업 취소

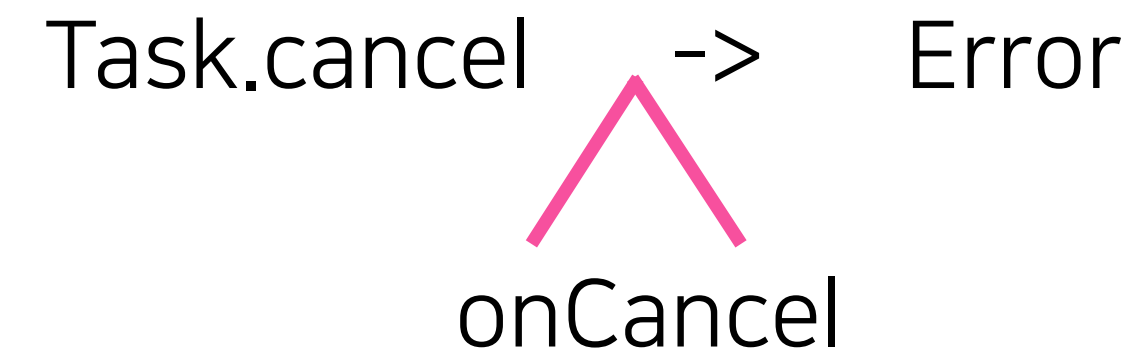


```
private var overviewTask: Task<CoinOverviewDT0, Error>?  
private var weeklyTask: Task<CoinWeeklyDT0, Error>?  
private var todayTask: Task<CoinTodayNewsDT0, Error>?
```

```
func cancelOverview() { overviewTask?.cancel() }  
func cancelWeekly() { weeklyTask?.cancel() }  
func cancelToday() { todayTask?.cancel() }
```

작업 취소 withTaskCancellationHandler

Task가 취소될 때 실행할 클린업 동작을 등록



```
weeklyTask = Task { [weak self] in
    try await withTaskCancellationHandler(
        operation: {
            guard let self else { throw CancellationError() }
            return try await self.alanAPIService.fetchWeeklyTrends(for: self.coin)
        },
        onCancel: { [weak self] in
            self?.weekly = .cancel(.taskCancelled)
        },
        isolation: MainActor.shared
    )
}
```

실제 작업 실행

취소될 때 실행할 코드

onCancel이 실행될 스레드

AttributedString != String

문자열에 다양한 스타일(속성)을 입힐 수 있는 타입

폰트, 색상, 링크, 굵기, 밑줄

```
var overview = AttributedString()
```

```
guard let url = URL(string: urlString) else { ... }
```

```
let prefix = AttributedString("- 웹사이트: ") ..... 기본 텍스트
```

```
var link = AttributedString(url.host ?? urlString) ..... URL에서 host부분만 뽑아옴 (String?)
```

```
link.link = url ..... .link: link AttributedString을 클릭하면 url로 이동
```

```
link.foregroundColor = .aiCoAccent ..... 색상
```

```
link.underlineStyle = .single ..... 밑줄
```

```
overview.append(prefix)
```

```
overview.append(link)
```

```
overview.append(AttributedString("\n"))
```

```
return overview
```