

# Let'Swift 2023 Swift로 CrossPlatform하기

## Deep Dive into the unknown

Let'Swift Speaker Jihoon

# 프로필

Jihoon



안지훈 (Jihoon)

- 다양한 방식의 접근을 좋아하는 개발자
- 0년차 iOS 개발자
- 마이너한거 좋아함
- tmi) 곧 군대감

 [github.com/Jihoonahn](https://github.com/Jihoonahn)

 @jihoonahn



**Let'Swift 2023**  
Deep Dive into the unknown

## Contents

1. Swift로 Cross Platform을?
2. Swift CrossPlatform?
3. 무엇을 제공해주는지
4. Ready to play Swift CrossPlatform
5. Start Swift CrossPlatform
6. 예시 프로젝트 및 설명
7. 장단점 설명
8. 마무리

# Swift로 Cross Platform을?



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift로 Cross Platform을?

만들던 iOS 앱 프로젝트가  
안드로이드에서도 실행됐으면 좋겠는데..

그런데, 두개 다 만들었다고 해도,  
이걸 내가 두개 다 관리할 시간이 생길까?



# Swift로 Cross Platform을?

## CrossPlatform을 해야하나?



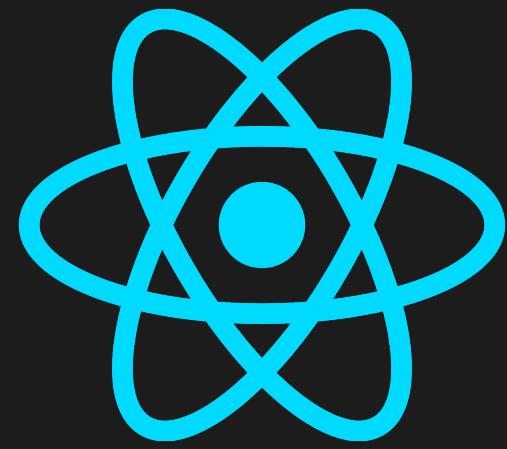
**Let'Swift 2023**  
Deep Dive into the unknown

# Swift로 Cross Platform을?



Kotlin Multiplatform

Kotlin



React Native

Javascript



Flutter

Dart



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift로 Cross Platform을?

RN이나 Flutter 배우면 기존에 Swift만큼의 숙련도를 낼 수 있을까..

다른 언어는 언제 배워..

새로운 IDE 익숙해지는거도 시간 걸릴텐데..



저거 두개 관리하면 너무 시간이 없어질거 같은데..

# Swift로 Cross Platform을?

기존 언어를 이용해서 학습에 필요한 리소스 최소화



iOS, Android 동시 개발을 통한 생산성 향상



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift로 Cross Platform 을?



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift CrossPlatform?



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift CrossPlatform?



[ S C A D E ]

## Scade

- Swift 5.x 지원 (현재 5.8까지 지원)
- 동일한 소스 코드를 사용하여 iOS & Android 동시 개발
- 순수 Swift 및 Swift Foundation과 Swift Dispatch를 사용하는 모든 라이브러리 사용 가능



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift CrossPlatform?

- 2017. 5. 5 Swift 3.1 지원, WebView 제어 지원, 메모리 개선, 유용성 개선, 애니메이션 API에 대한 수정....
- 2017. 7. 31 카메라 지원, 사이드바 제어, 마스터 페이지 제어....
- 2017. 11 Swift 4 및 Xcode 9 지원, Android 및 iOS 에 대한 Swift 기반 지원, 프로그래밍 방식 UI 생성, Testflight 호환 기능...등

<https://docs.scade.io/docs/changelog>

■ ■ ■

- 2023. 8 (최근 업데이트: Version 2.3.1 Beta)

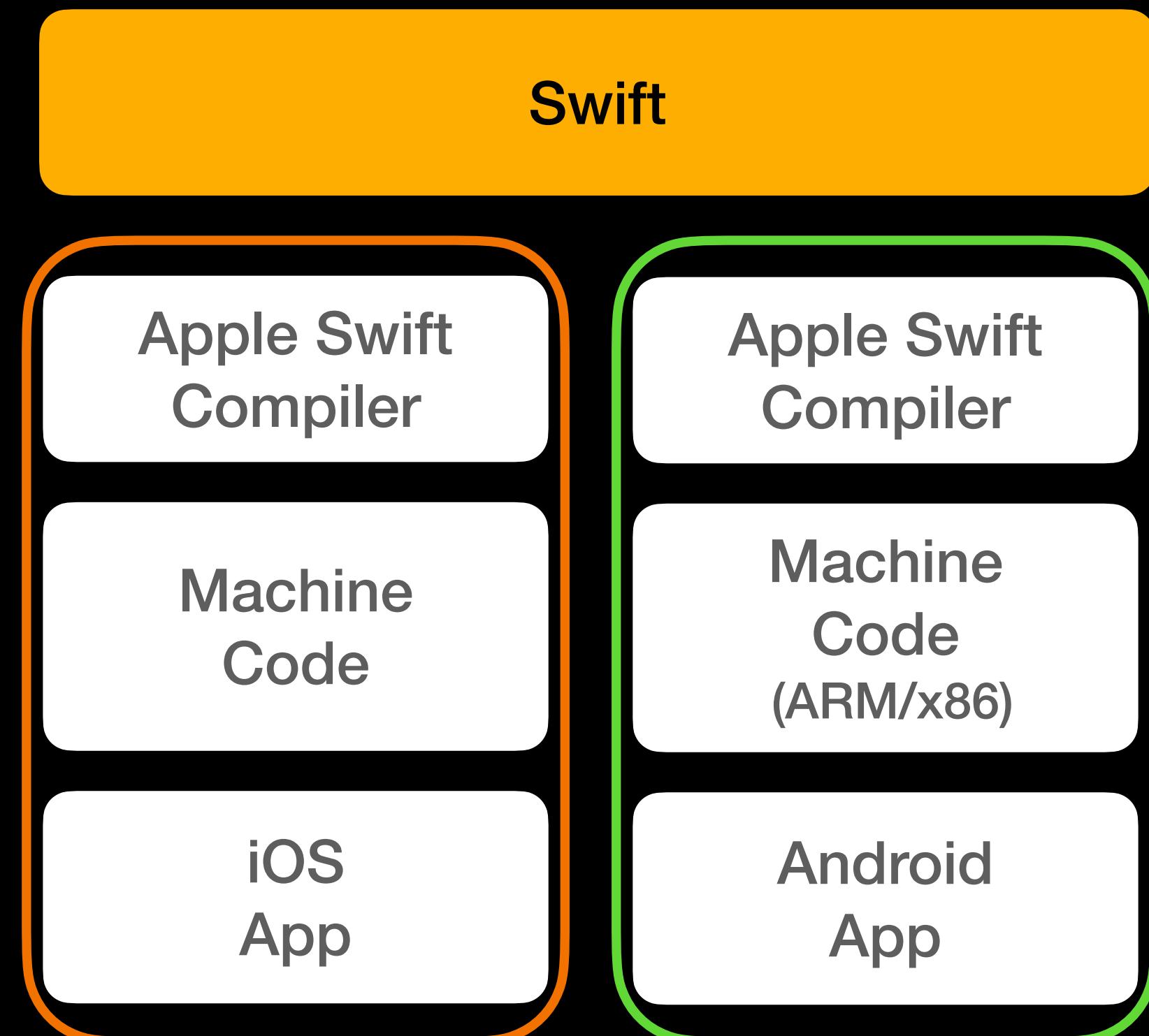
최근 지원: Swift 5.8 지원, 맞춤형 LSP, 컬렉션 보기 UI 위젯 지원, Android용 URLSession에 비동기 기능 지원, scd 명령줄 빌드 도구 추가

# 무엇을 제공해주는지



**Let'Swift 2023**  
Deep Dive into the unknown

# 무엇을 제공해주는지



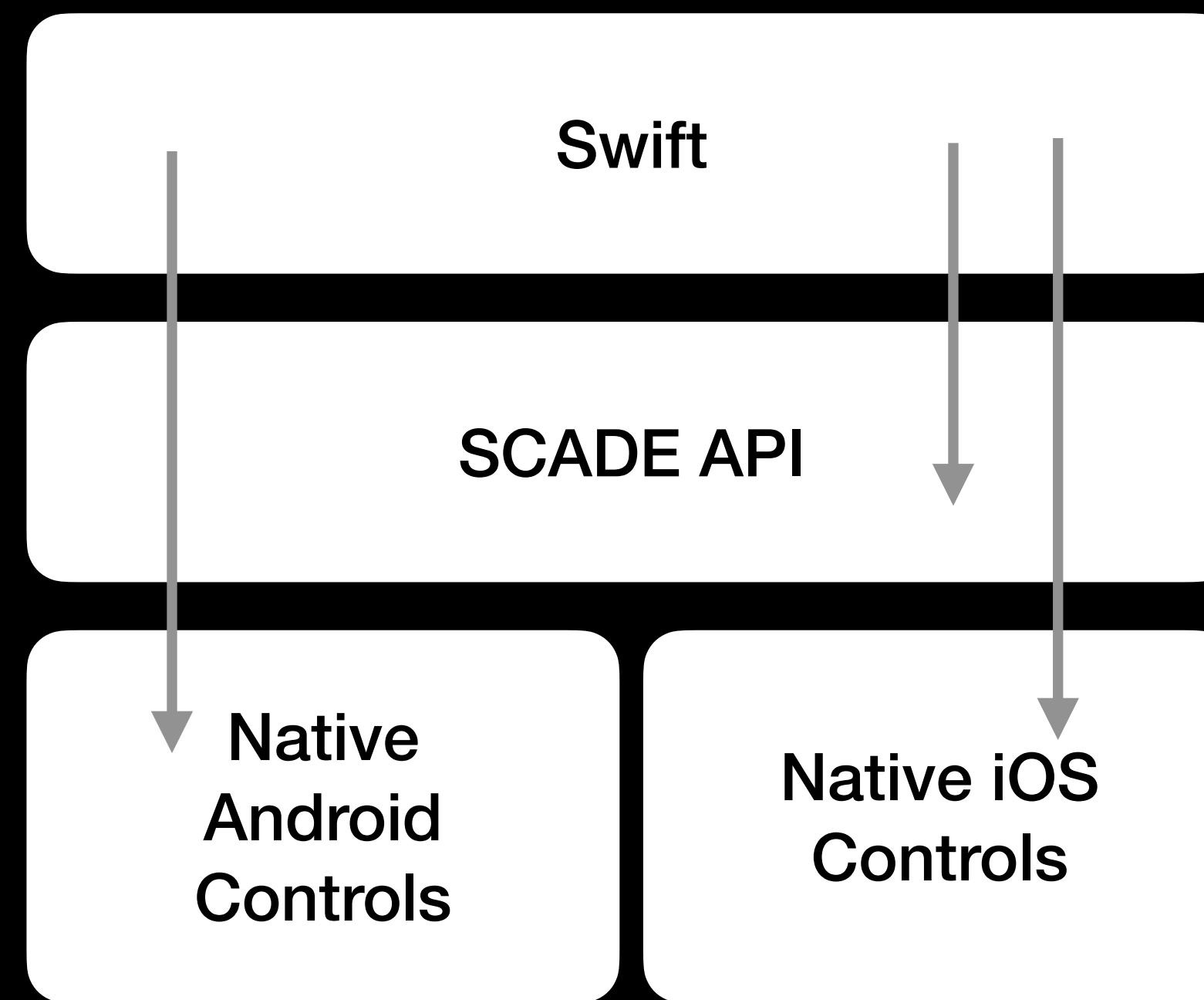
## Swift Compiler for Android

- Swift 5.4 소스코드를 가져와 Android 플랫폼용 기본 바이너리 이미지로 컴파일
- x86 과 ARM 모두 32bit 및 64bit 플랫폼 지원

<https://docs.scade.io/docs/how-scade-works>



# 무엇을 제공해주는지



## Crystal (High Performance Graphics Engine)

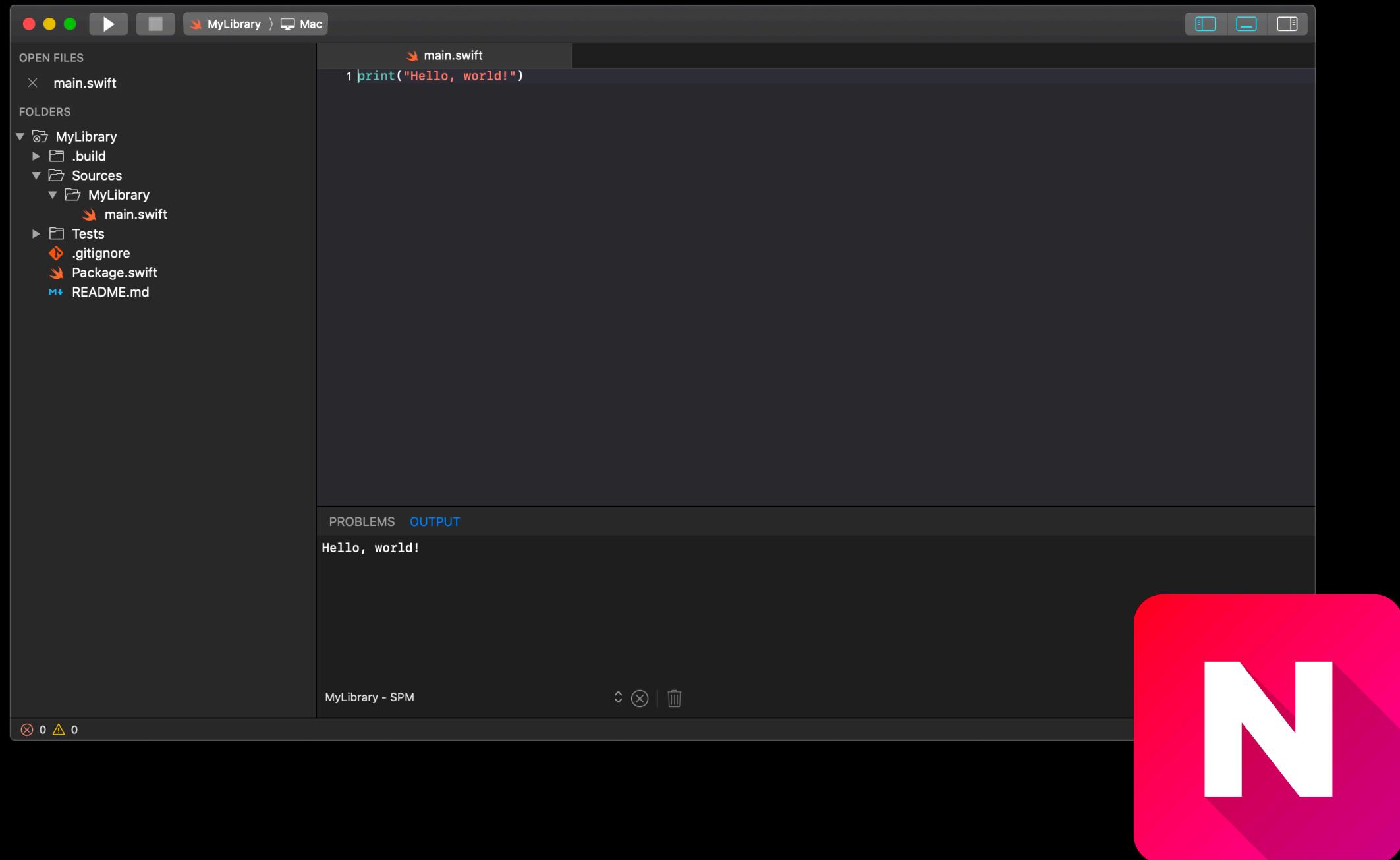
- Vector Graphic 과 Native OS UI의 통합
- Crystal은 Native Control(텍스트 필드, 키보드, 지도, 카메라.. 등) 을 사용 가능

<https://docs.scade.io/docs/how-scade-works>



**Let'Swift 2023**  
Deep Dive into the unknown

# 무엇을 제공해주는지



## Nimble (SCADE IDE)

- 자체 Scade IDE 제공 (오픈소스)
- Swift Package Manager 프로젝트 생성 가능
- LSP를 통한 자동완성 기능, 진단 기능 제공
- Scade Simulator 지원

<https://github.com/scade-platform/Nimble>



**Let'Swift 2023**  
Deep Dive into the unknown

# 무엇을 제공해주는지

The screenshot shows a code editor window for a file named \*main.swift. The code contains:

```
1 import Foundation
2
3 let k: Int
```

A code completion dropdown is open, listing various integer types. The item 'Int Int' is highlighted in blue. Other items include:

- S Int Int
- S Int8 Int8
- S Int16 Int16
- S Int32 Int32
- S Int64 Int64
- T Int IntegerLiteralType
- S Int1Text Int1Text
- T Int intmax\_t

Below the code editor, a red box highlights a warning message:

① Cannot find type 'SomeProtocol' in scope

At the bottom, there is a warning message about an unused immutable value:

⚠ Initialization of immutable value 'str' was never used; consider replacing with assignment to '\_' or removing it  
Replace 'let str' with '\_'

Fix

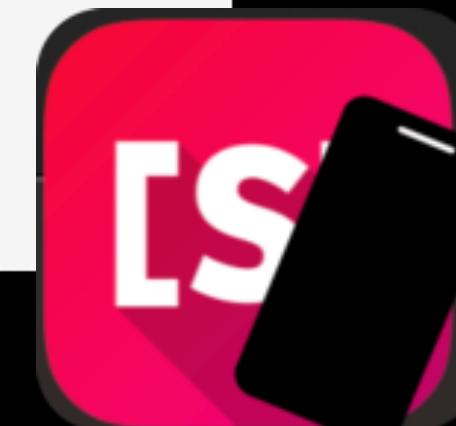
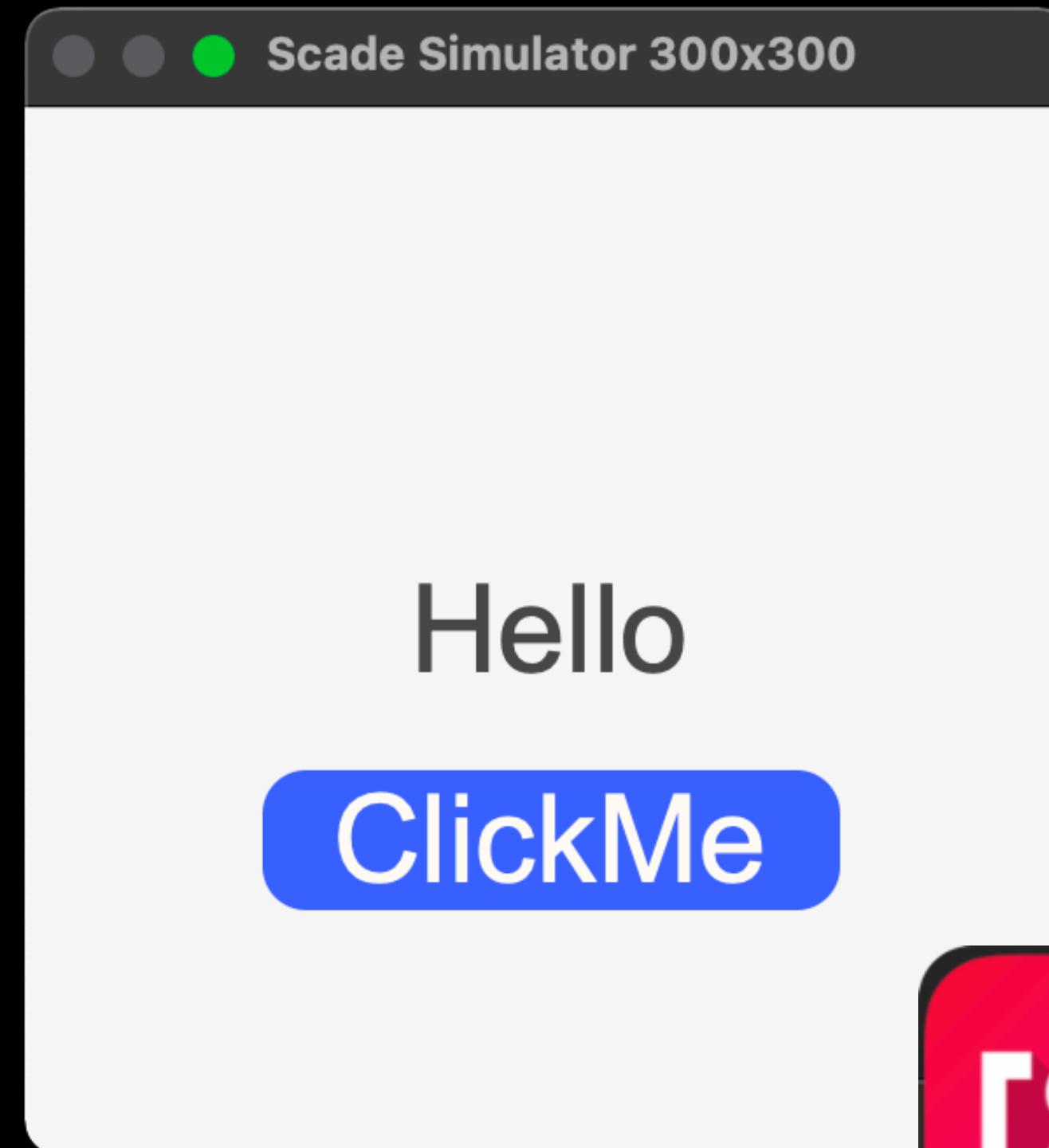
## Nimble (SCADE IDE)

- 자체 Scade IDE 제공 (오픈소스)
- Swift Package Manager 프로젝트 생성 가능
- LSP를 통한 자동완성 기능, 진단 기능 제공
- Scade Simulator 지원

<https://github.com/scade-platform/Nimble>



# 무엇을 제공해주는지



## Nimble (SCADE IDE)

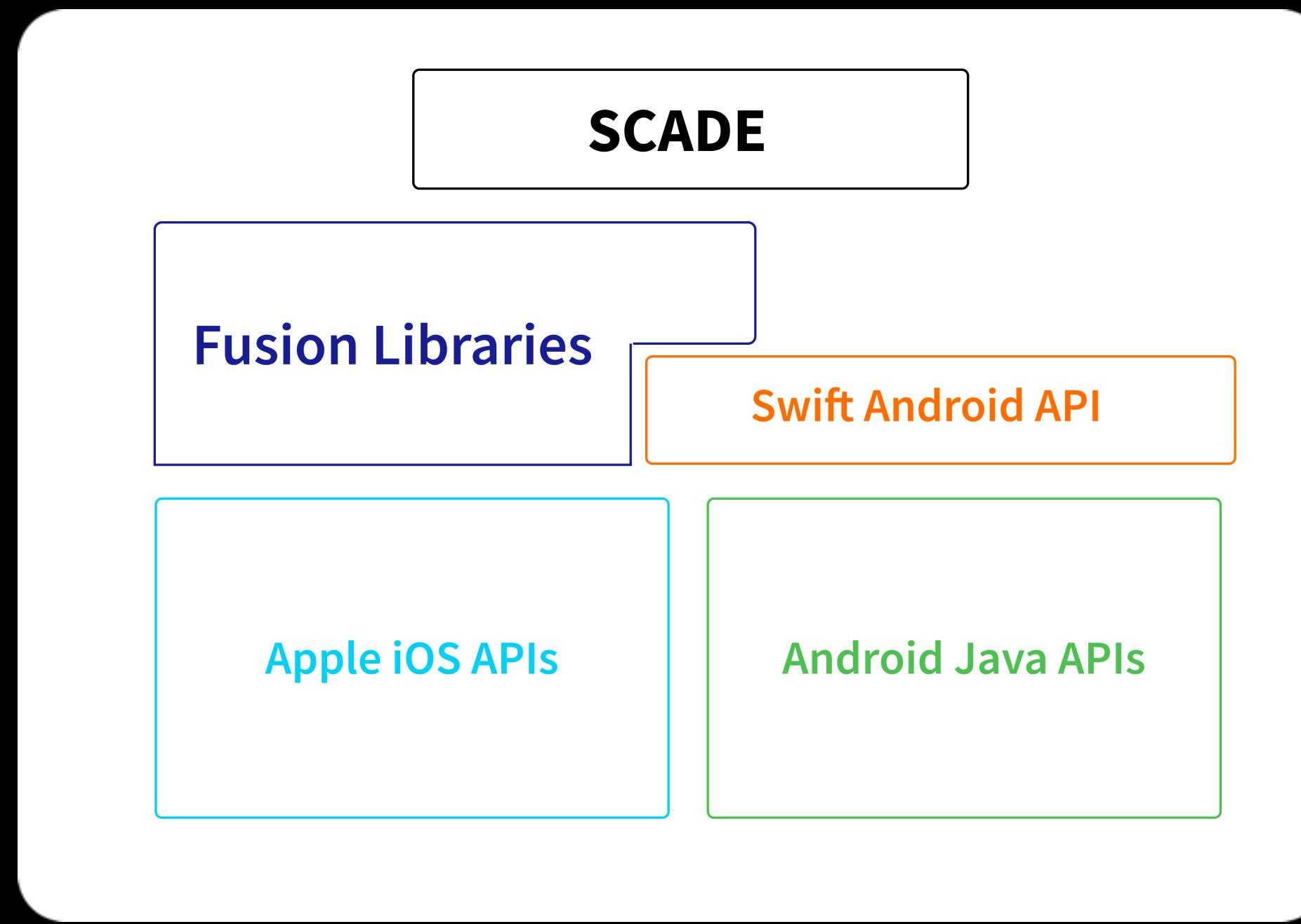
- 자체 Scade IDE 제공 (오픈소스)
- Swift Package Manager 프로젝트 생성 가능
- LSP를 통한 자동완성 기능, 진단 기능 제공
- Scade Simulator 지원

<https://github.com/scade-platform/Nimble>



**Let'Swift 2023**  
Deep Dive into the unknown

# 무엇을 제공해주는지



Fusion Architecture

## Fusion

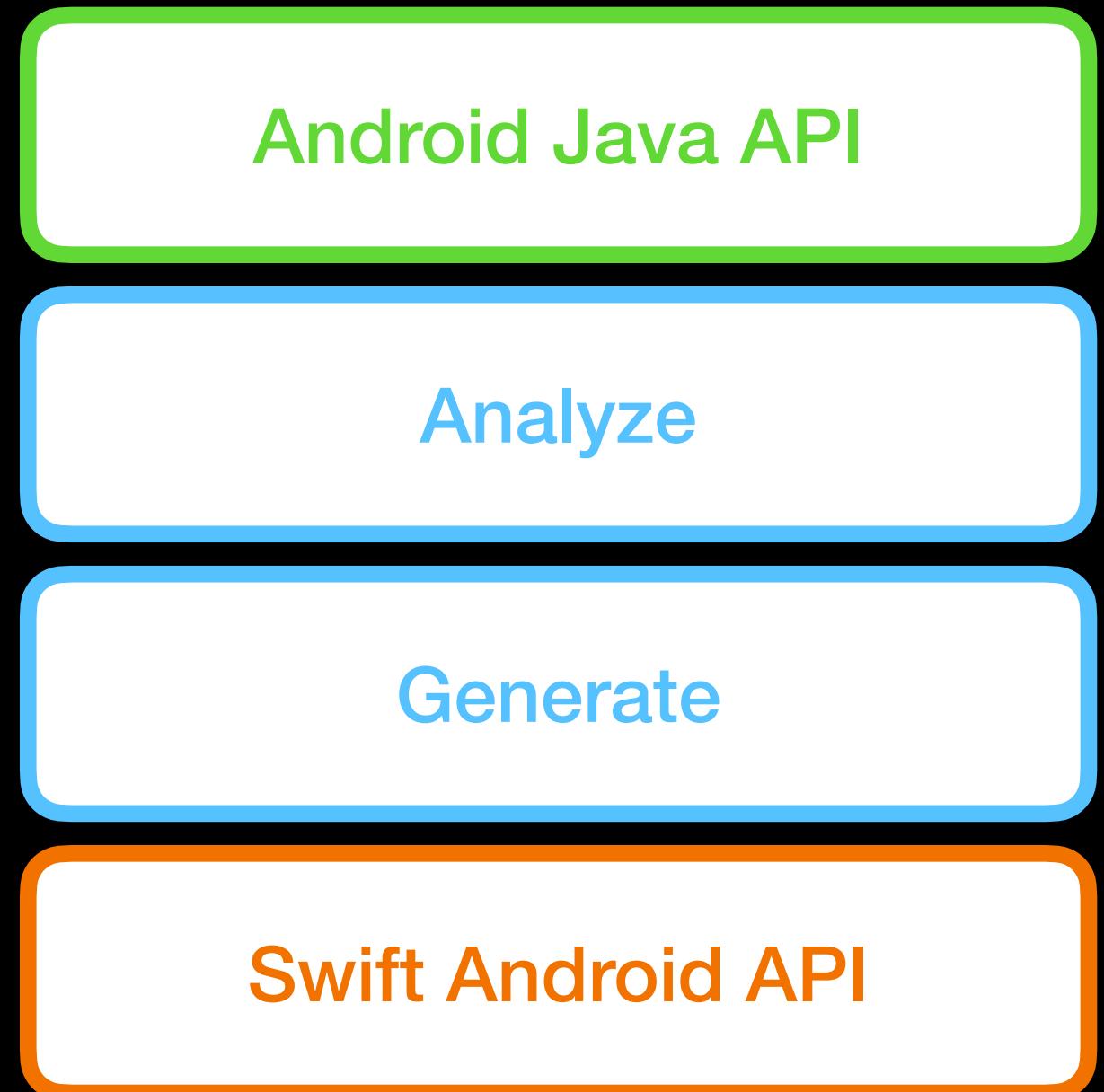
- Swift 코드를 사용하여 Android API를 호출 가능
- Fusion Library 및 API에 접근하여 크로스 플랫폼 앱을 쉽게 구축

<https://docs.scade.io/docs/fusionintroduction>



**Let'Swift 2023**  
Deep Dive into the unknown

# 무엇을 제공해주는지



Fusion

- Swift 코드를 사용하여 Android API를 호출 가능
- Fusion Library 및 API에 접근하여 크로스 플랫폼 앱을 쉽게 구축

Fusion Generation Process

<https://docs.scade.io/docs/fusionintroduction>



**Let'Swift 2023**  
Deep Dive into the unknown

# 무엇을 제공해주는지

S.No	Library Name	Repository	Description
1	FusionLocation	<a href="https://github.com/scade-platform/FusionLocation">https://github.com/scade-platform/FusionLocation</a>	Android & iOS Geolocation impl
2	FusionNFC	<a href="https://github.com/scade-platform/FusionNFC">https://github.com/scade-platform/FusionNFC</a>	NFC read/write impl for iOS and Android
3	FusionMedia	<a href="https://github.com/scade-platform/FusionMedia">https://github.com/scade-platform/FusionMedia</a>	Media player for the iOS and Android platform
4	FusionBluetooth	<a href="https://github.com/scade-platform/FusionBluetooth">https://github.com/scade-platform/FusionBluetooth</a>	Bluetooth impl for iOS and Android
5	FusionLocalAuth	<a href="https://github.com/scade-platform/FusionLocalAuth">https://github.com/scade-platform/FusionLocalAuth</a>	Local authentication using biometrics for iOS and Android
6	FusionQR	<a href="https://github.com/scade-platform/FusionCamera">https://github.com/scade-platform/FusionCamera</a>	QR code detection using Camera for iOS and Android

Fusion Libraries

## Fusion

- Swift 코드를 사용하여 Android API를 호출 가능
- Fusion Library 및 API에 접근하여 크로스 플랫폼 앱을 쉽게 구축

<https://docs.scade.io/docs/fusionintroduction>



**Let'Swift 2023**  
Deep Dive into the unknown

# Swift Compiler for Android

[ S C A D E ]



Nimble



Crystal



Fusion



Let'Swift 2023  
Deep Dive into the unknown

# Ready to play Swift CrossPlatform



**Let'Swift 2023**  
Deep Dive into the unknown

# Ready to play Swift CrossPlatform

## Download List

- Xcode (현재는 14.x 버전 까지 지원)
- AndroidStudio
- Scade



**Let'Swift 2023**  
Deep Dive into the unknown

# Ready to play Swift CrossPlatform

Native Cross Platform App Development with Swift

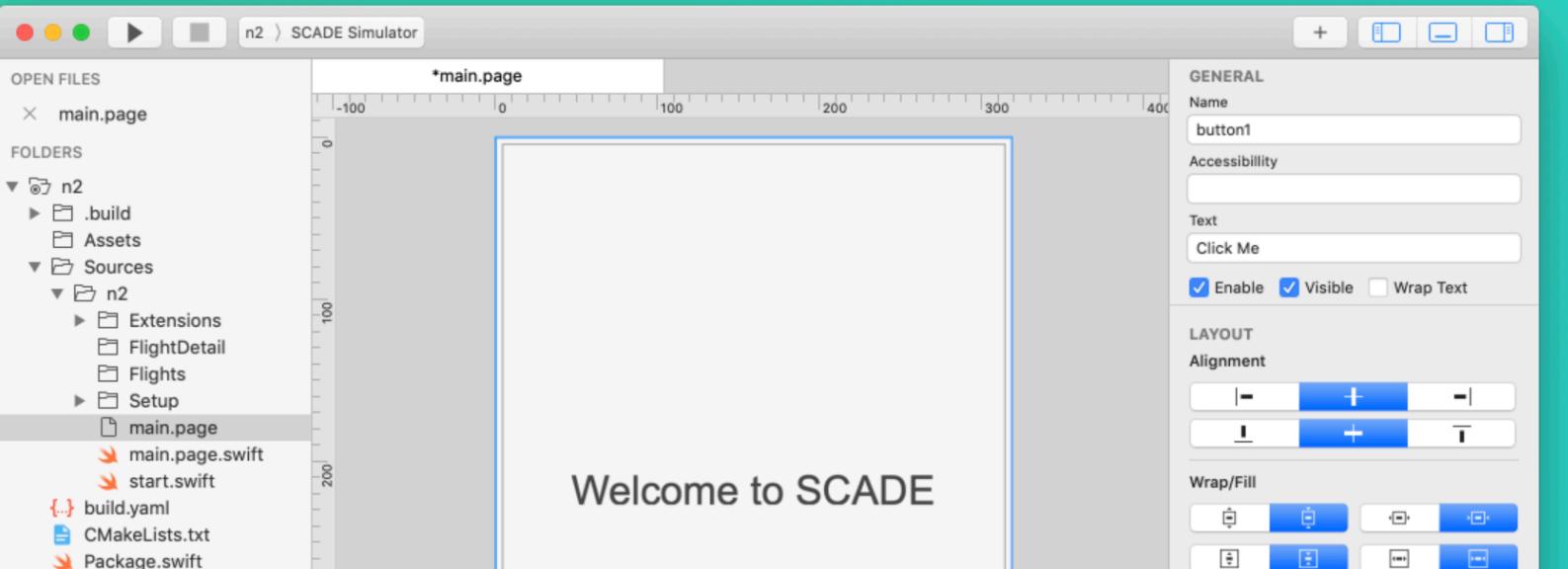
Download SCADE 2

Current version 2.3.1 (19 August 2023)

[Changes to SCADE software](#)

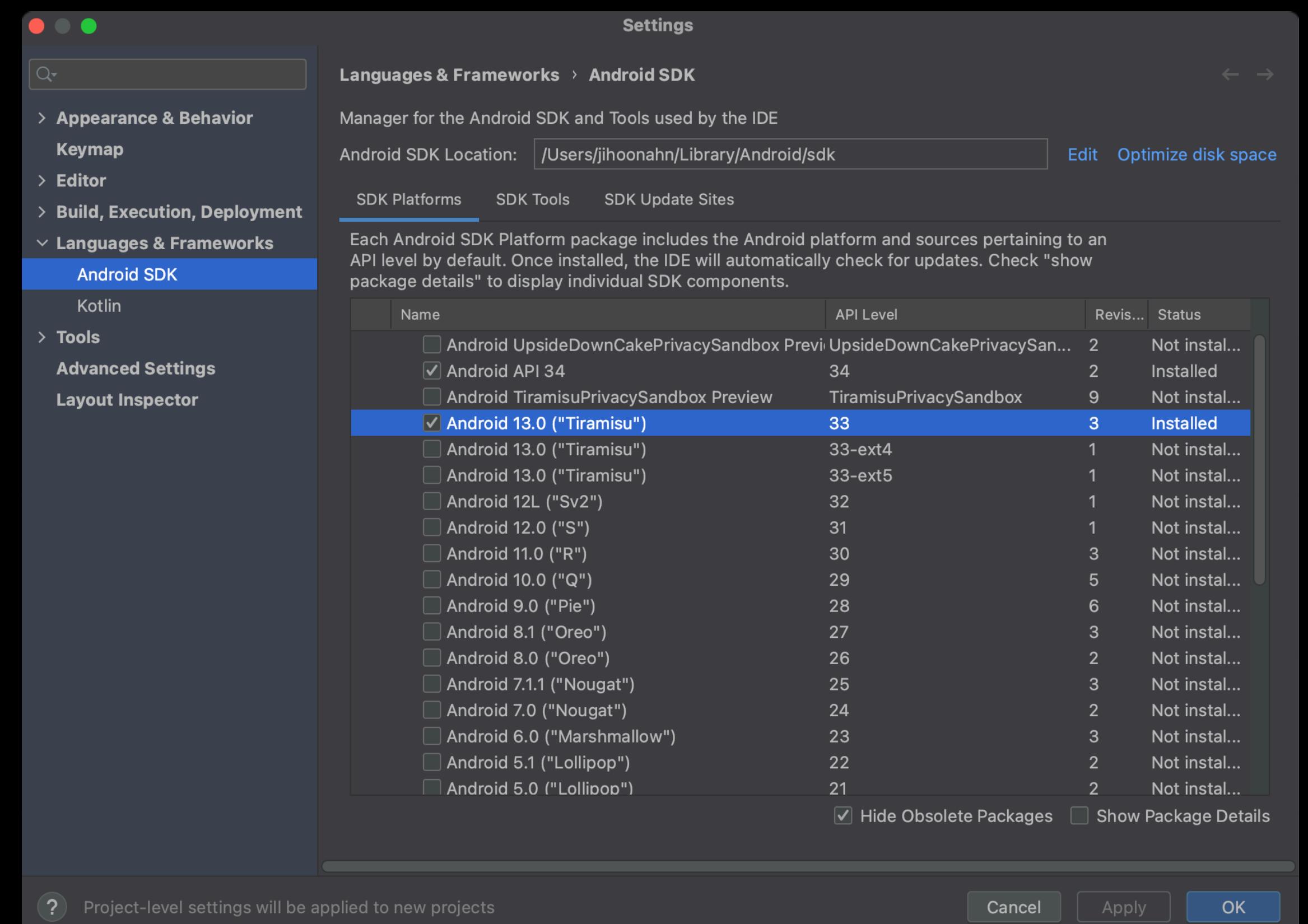
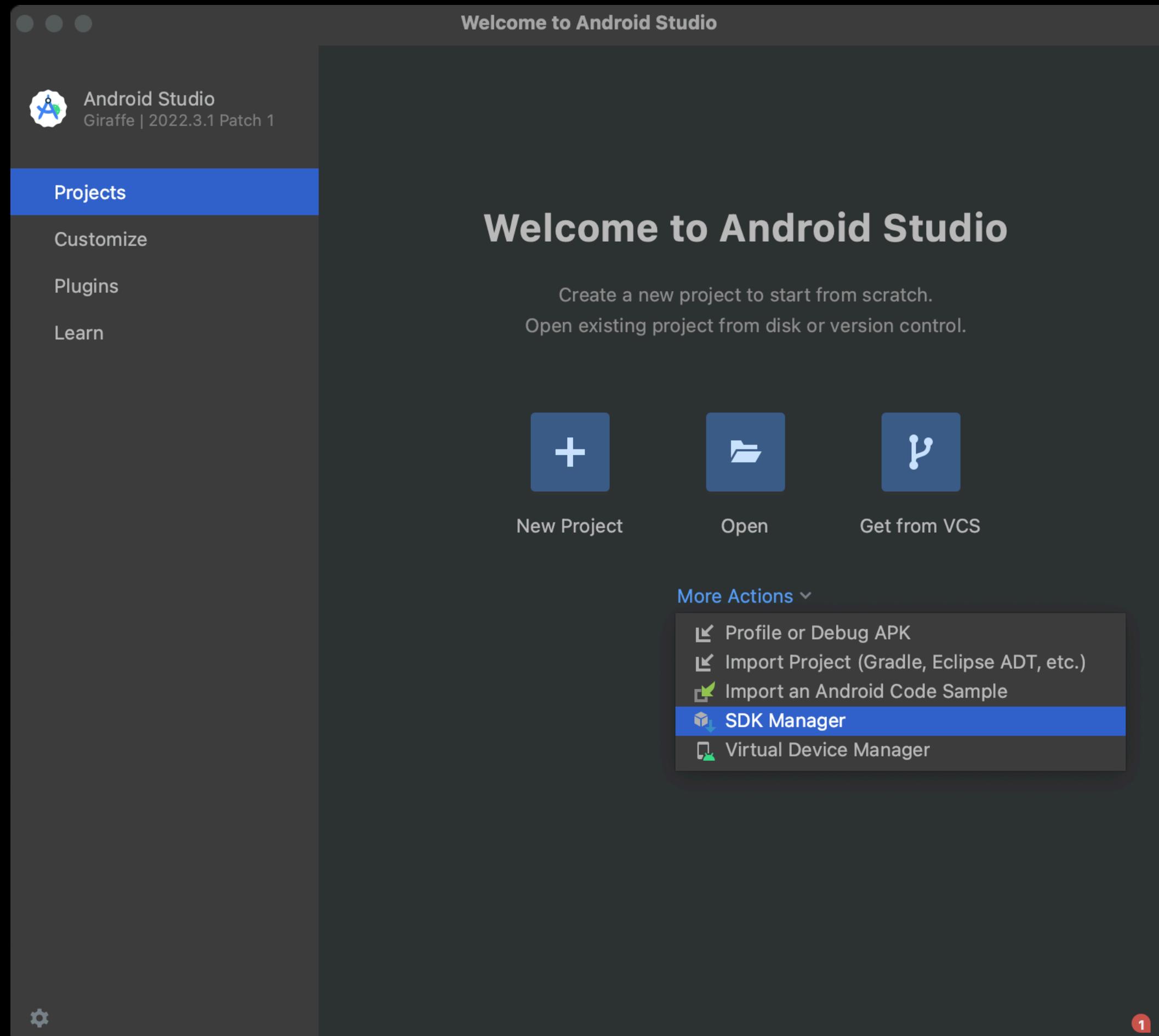
Subscribe me to newsletter optional@email.com

[Download Now](#)

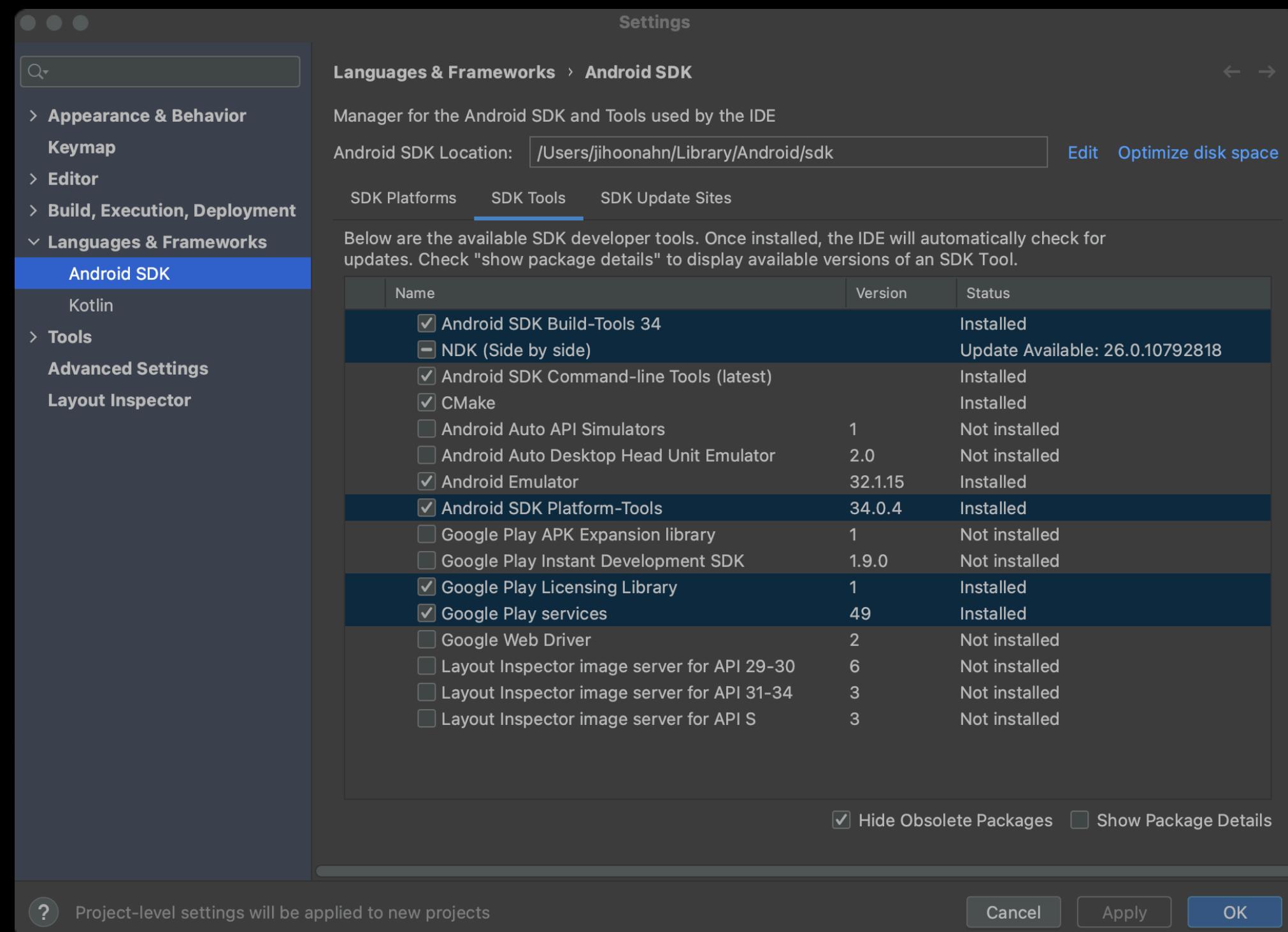


<https://www.scade.io/download/>

# Ready to play Swift CrossPlatform



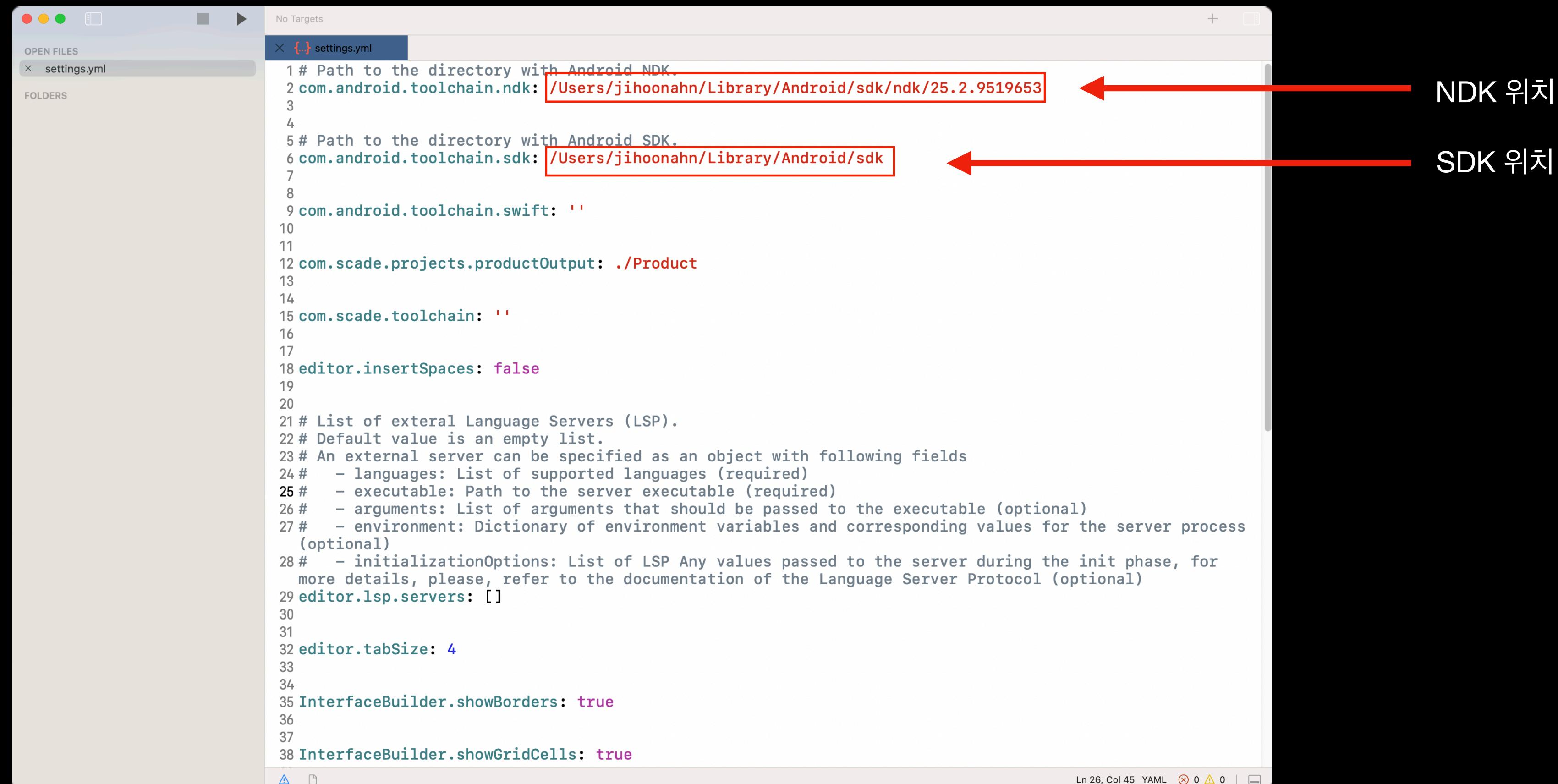
# Ready to play Swift CrossPlatform



## Install SDK Tools

- Android SDK Platform-Tools
- Android SDK Build-Tools
- NDK (Side by side)
- Google Play services
- Google Play Licensing Library

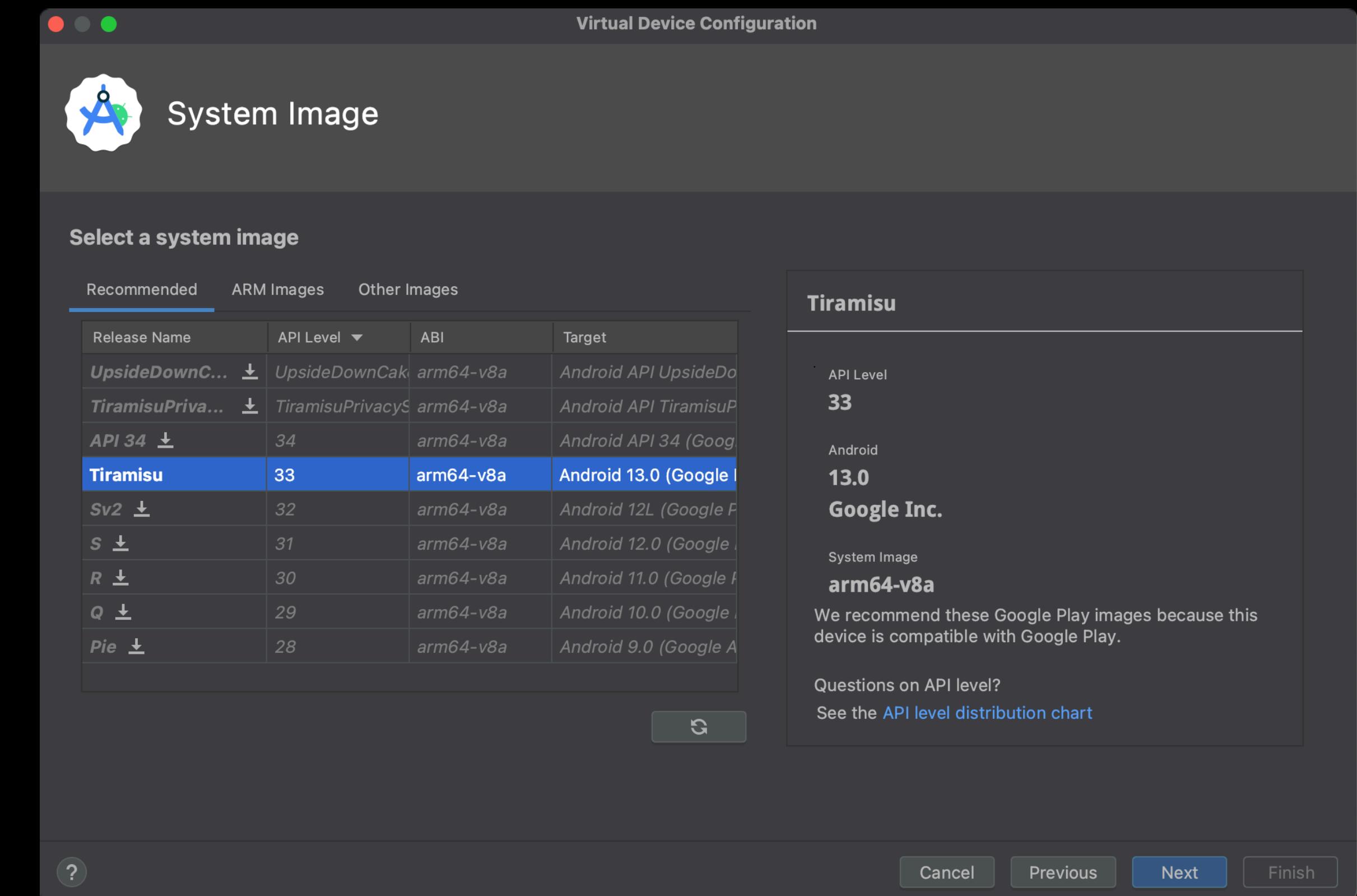
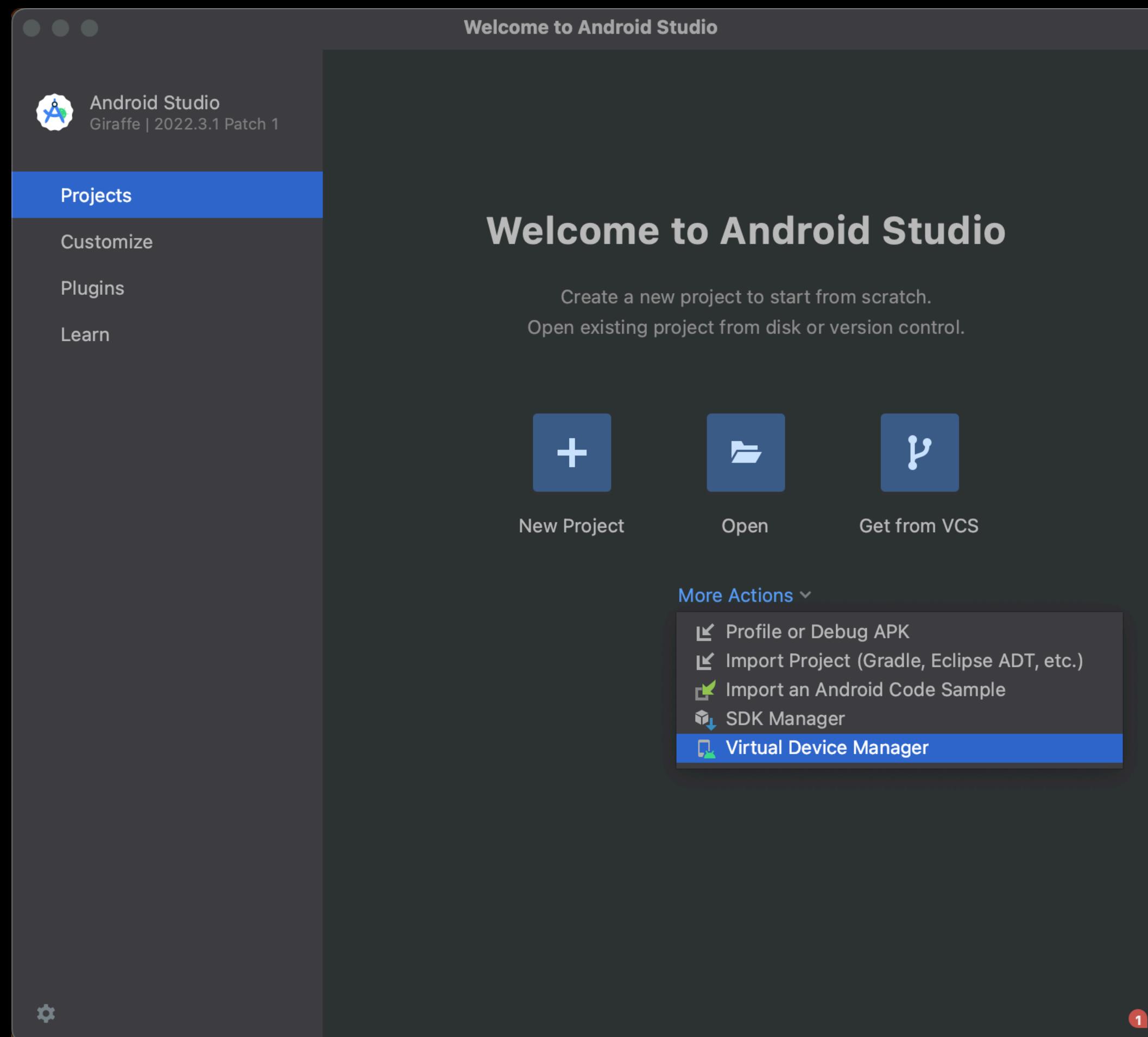
# Ready to play Swift CrossPlatform



```
No Targets  
X {..} settings.yml  
OPEN FILES  
X settings.yml  
FOLDERS  
1 # Path to the directory with Android NDK.  
2 com.android.toolchain.ndk: /Users/jihoonahn/Library/Android/sdk/ndk/25.2.9519653 ← NDK 위치  
3  
4  
5 # Path to the directory with Android SDK.  
6 com.android.toolchain.sdk: /Users/jihoonahn/Library/Android/sdk ← SDK 위치  
7  
8  
9 com.android.toolchain.swift: ''  
10  
11  
12 com.scade.projects.productOutput: ./Product  
13  
14  
15 com.scade.toolchain: ''  
16  
17  
18 editor.insertSpaces: false  
19  
20  
21 # List of external Language Servers (LSP).  
22 # Default value is an empty list.  
23 # An external server can be specified as an object with following fields  
24 #   - languages: List of supported languages (required)  
25 #   - executable: Path to the server executable (required)  
26 #   - arguments: List of arguments that should be passed to the executable (optional)  
27 #   - environment: Dictionary of environment variables and corresponding values for the server process  
     (optional)  
28 #   - initializationOptions: List of LSP Any values passed to the server during the init phase, for  
     more details, please, refer to the documentation of the Language Server Protocol (optional)  
29 editor.lsp.servers: []  
30  
31  
32 editor.tabSize: 4  
33  
34  
35 InterfaceBuilder.showBorders: true  
36  
37  
38 InterfaceBuilder.showGridCells: true
```



# Ready to play Swift CrossPlatform



# Ready to play Swift CrossPlatform



Scade 사용 준비 끝!



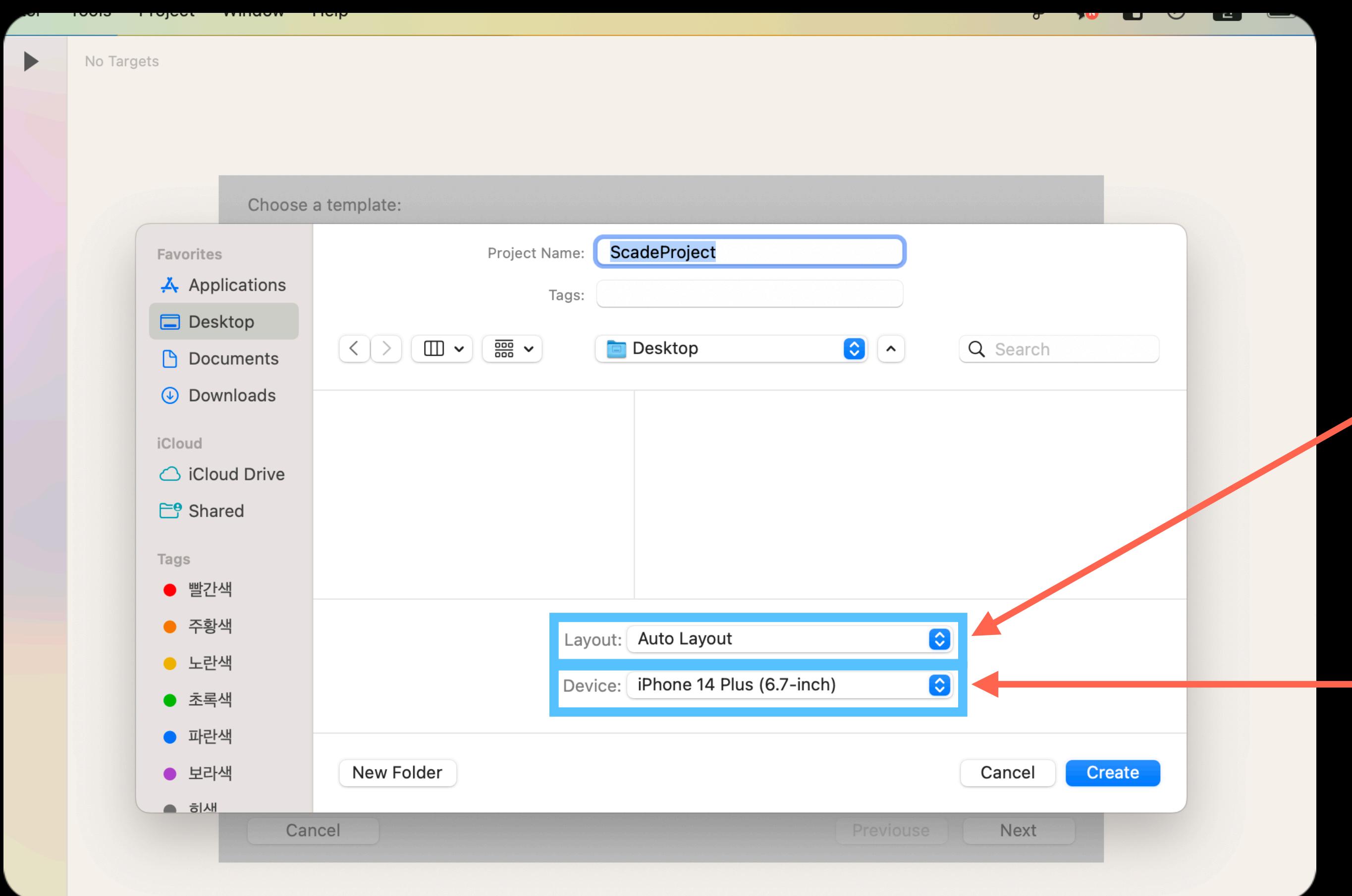
Let'Swift 2023  
Deep Dive into the unknown

# Start Swift CrossPlatform



**Let'Swift 2023**  
Deep Dive into the unknown

# Start Swift CrossPlatform



기본 레이아웃 관리자를 선택. ( AutoLayout/GridLayout )

기본 디바이스 크기



Let'Swift 2023  
Deep Dive into the unknown

# Start Swift CrossPlatform

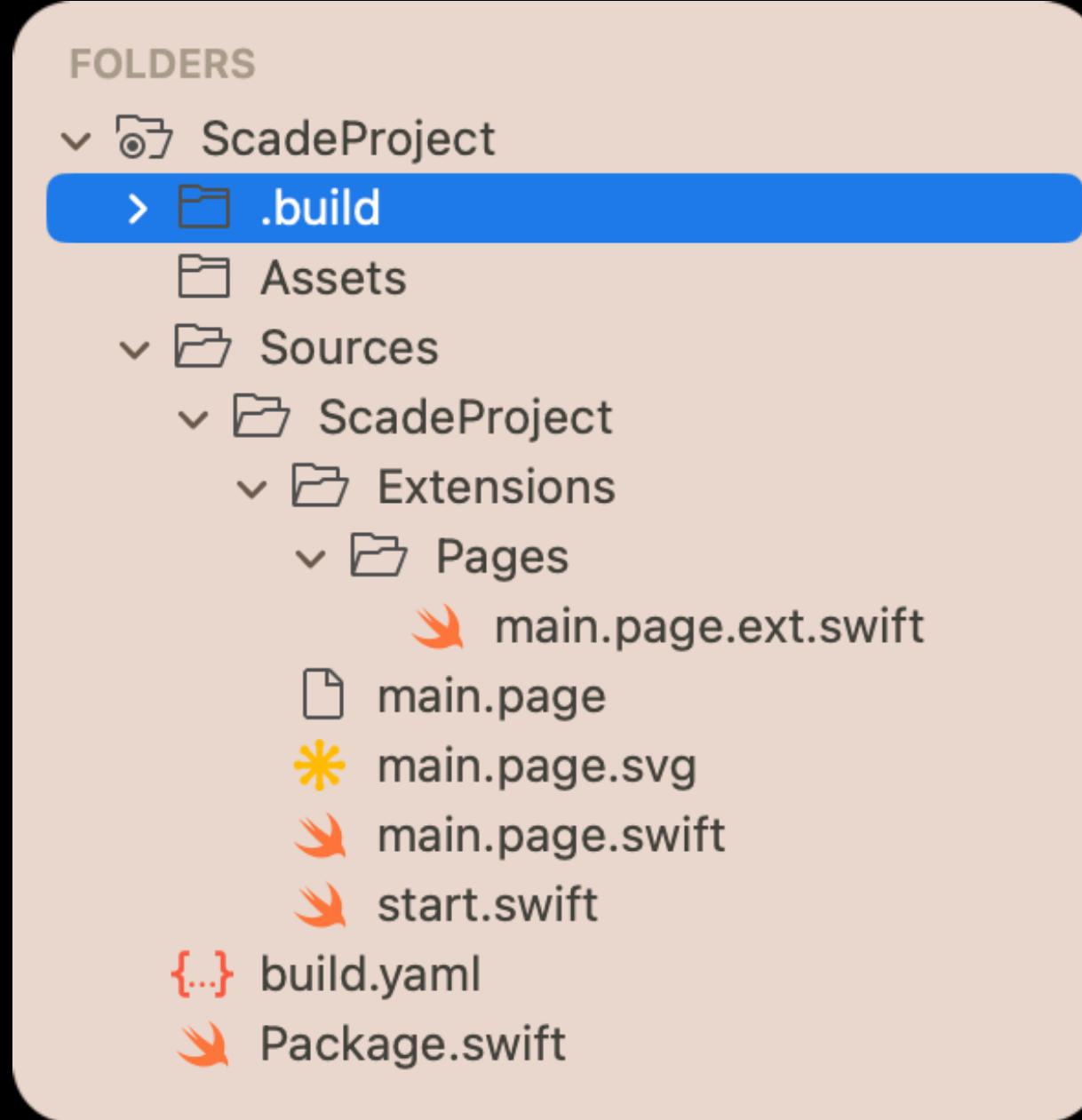
FOLDERS	
▼	ScadeProject
└	Assets
▼	Sources
└	ScadeProject
└	main.page
*	main.page.svg
*	main.page.swift
*	start.swift
{...}	build.yaml
*	Package.swift

Before Build

File & Folder	Description
Assets	모바일 앱 이미지나, 아이콘들을 저장하는 단일 폴더
main.page	페이지 빌더라고도 불립니다. 메인 페이지를 시각적으로 표현한 것으로 Storyboard와 비슷한 역할을 합니다.
main.page.swift	메인 페이지 소스가 포함된 Swift 파일입니다.
start.swift	어플리케이션의 시작점입니다. iOS에서 App Delegate와 같은 기능이 포함되어 있습니다.
Package.swift	manifest file이라고 알려져 있으며, 프로젝트에 필요한 종속성을 지정할 수 있습니다.
build.yaml	Android와 iOS의 특정 설정을 위한 파일입니다. 리소스, 권한 등등 build 파일에서 설정할 수 있습니다.



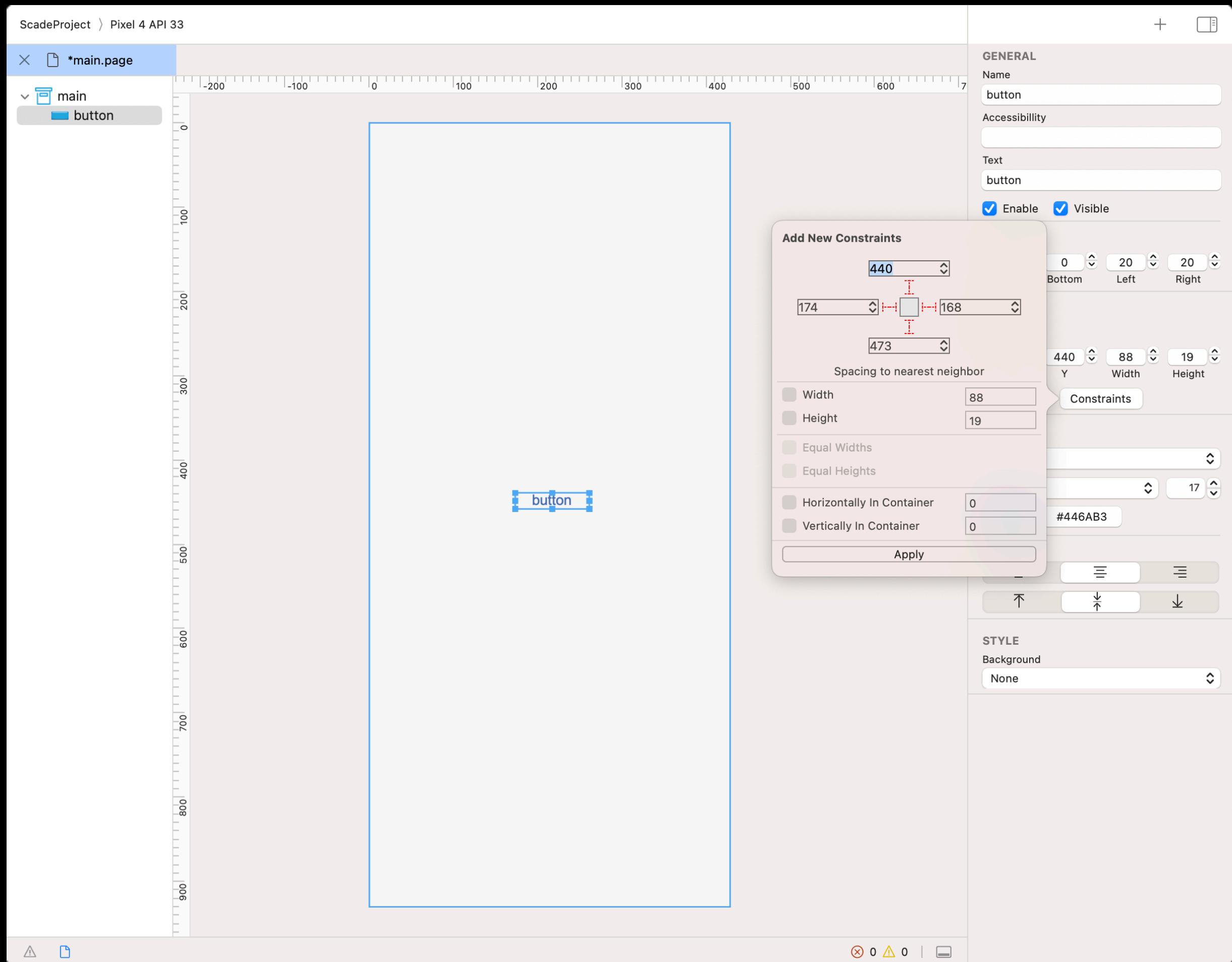
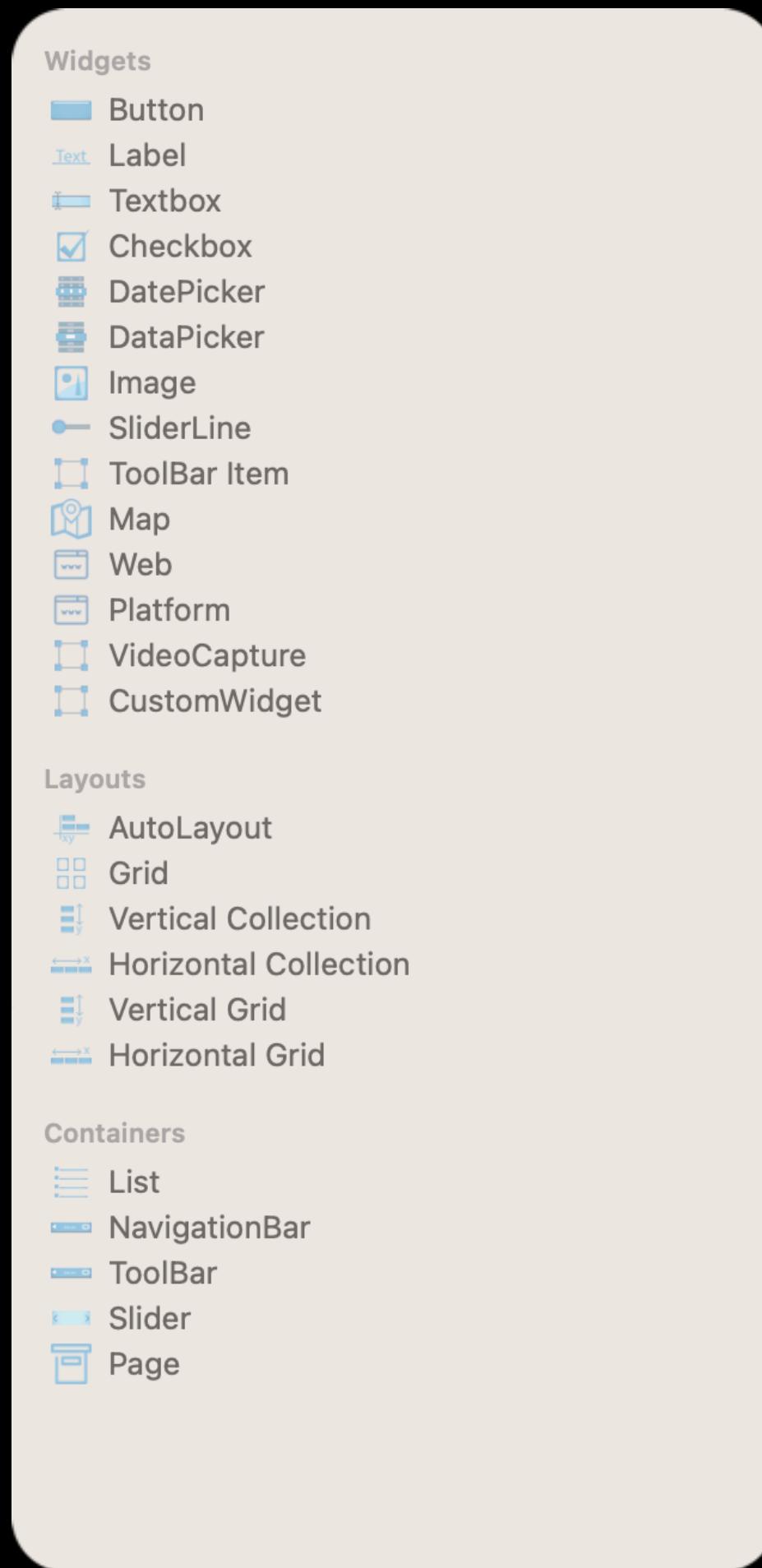
# Start Swift CrossPlatform



File & Folder	Description
<b>Assets</b>	모바일 앱 이미지나, 아이콘들을 저장하는 단일 폴더
<b>.build</b>	컴파일러에서 컴파일된 산출물을 .build 폴더에 저장합니다.
<b>main.page</b>	페이지 빌더라고도 불립니다. 메인 페이지를 시각적으로 표현한 것으로 Storyboard와 비슷한 역할을 합니다.
<b>main.page.swift</b>	메인 페이지 소스가 포함된 Swift 파일입니다.
<b>start.swift</b>	어플리케이션의 시작점입니다. iOS에서 App Delegate와 같은 기능이 포함되어 있습니다.
<b>Extensions/Pages</b>	페이지 빌더가 작성되거나, 수정되었을 때, IDE에서 코드를 생성해서 하위 페이지에 저장합니다.
<b>Package.swift</b>	manifest file이라고 알려져 있으며, 프로젝트에 필요한 종속성을 지정할 수 있습니다.
<b>build.yaml</b>	Android와 iOS의 특정 설정을 위한 파일입니다. 리소스, 권한 등등 build 파일에서 설정할 수 있습니다.
<b>Product</b>	컴파일러에서 생성된 IPA 파일, APK 파일을 폴더에 저장합니다.

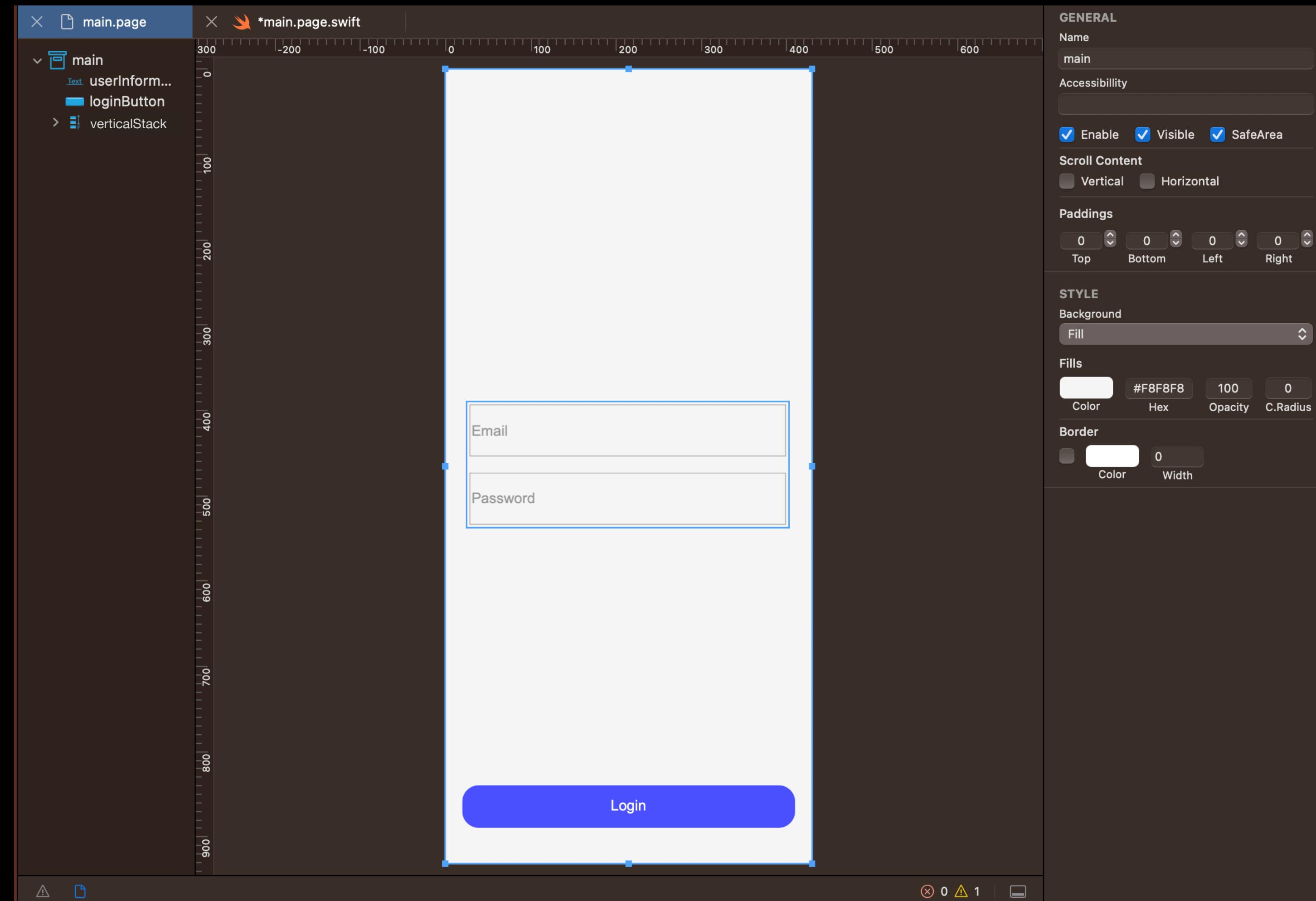


# Start Swift CrossPlatform



**Let'Swift 2023**  
Deep Dive into the unknown

# Start Swift CrossPlatform



**Let'Swift 2023**  
Deep Dive into the unknown

# Start Swift CrossPlatform

```
import ScadeKit

extension MainPageAdapter {
    var userInformationLabel: SCDWidgetsLabel {
        return self.page?.getWidgetByName("userInformationLabel") as! SCDWidgetsLabel
    }

    var loginButton: SCDWidgetsButton {
        return self.page?.getWidgetByName("loginButton") as! SCDWidgetsButton
    }

    var verticalStack: SCDWidgetsListView {
        return self.page?.getWidgetByName("verticalStack") as! SCDWidgetsListView
    }

    var emailTextfield: SCDWidgetsTextbox {
        return self.page?.getWidgetByName("emailTextfield") as! SCDWidgetsTextbox
    }

    var passwordTextfield: SCDWidgetsTextbox {
        return self.page?.getWidgetByName("passwordTextfield") as! SCDWidgetsTextbox
    }
}
```

Sources\\*(ProjectName)\Extensions\Pages\main.page.ext.swift



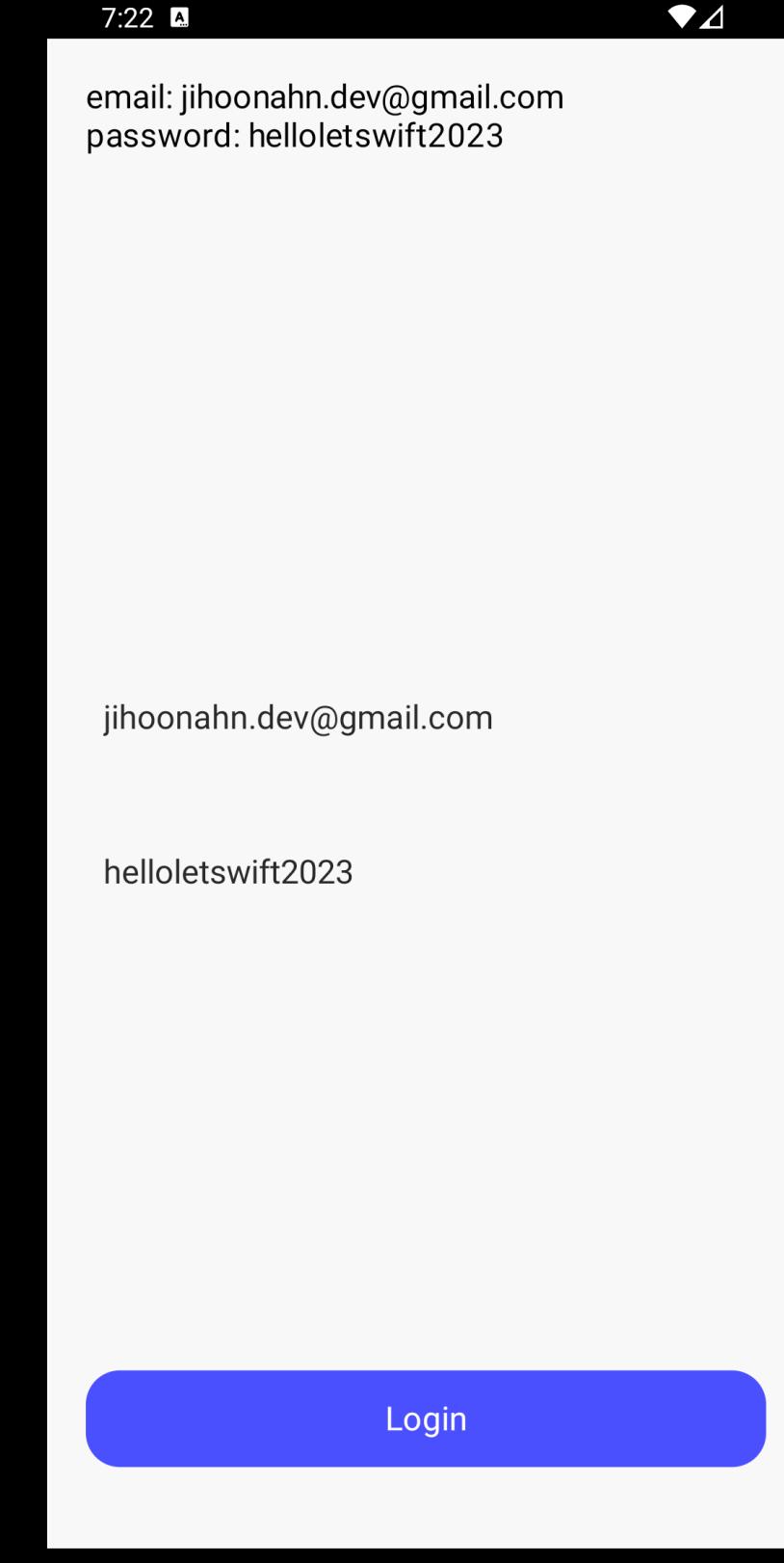
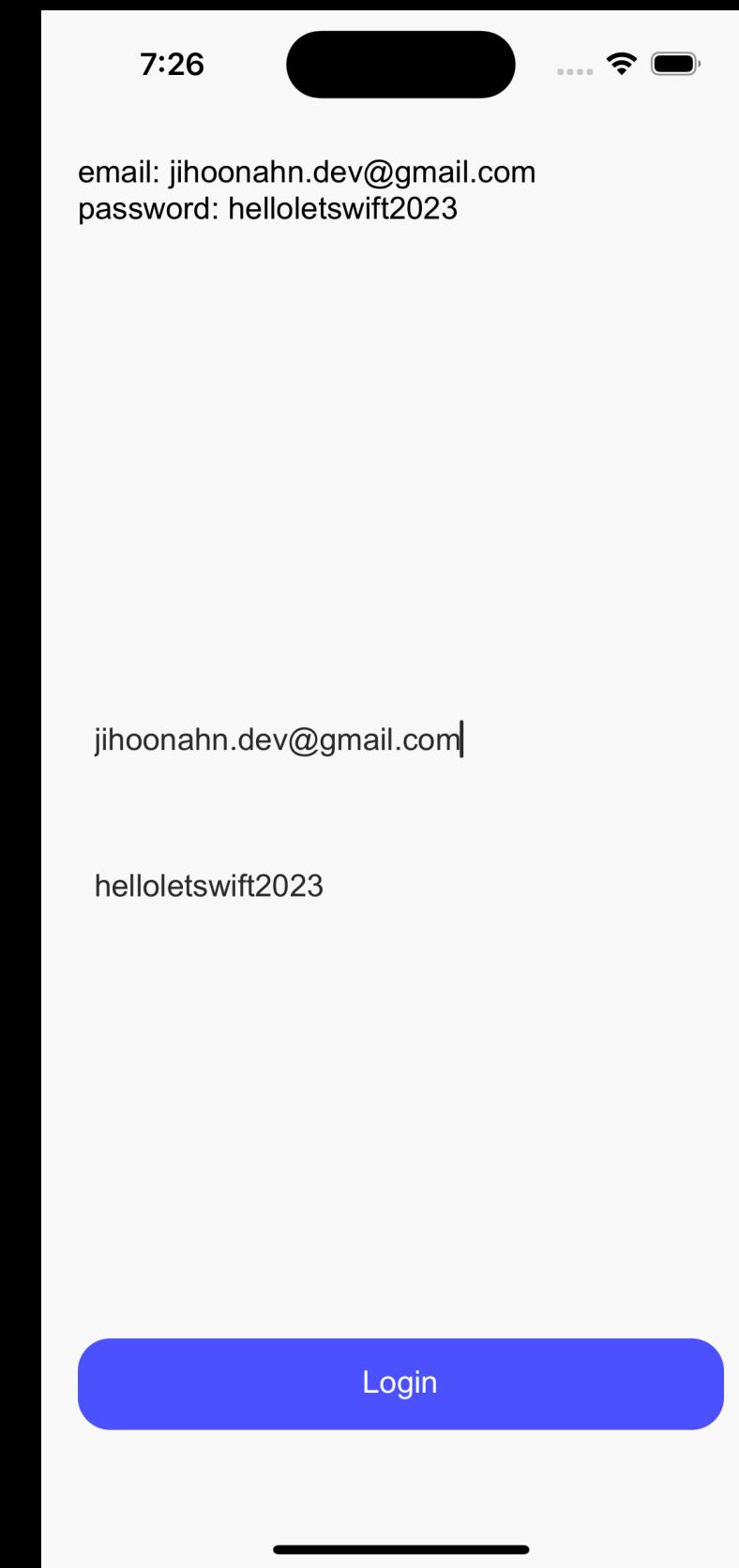
**Let'Swift 2023**  
Deep Dive into the unknown

# Start Swift CrossPlatform

```
import ScadeKit

class MainPageAdapter: SCDLatticePageAdapter {

    // page adapter initialization
    override func load(_ path: String) {
        super.load(path)
        loginButton.onClick { [self] _ in
            userInformationLabel.text = """
            email: \(emailTextfield.text)
            password: \(passwordTextfield.text)
            """
        }
    }
}
```



# 예시 프로젝트 및 코드 설명



**Let'Swift 2023**  
Deep Dive into the unknown

# 예시 프로젝트 및 코드 설명

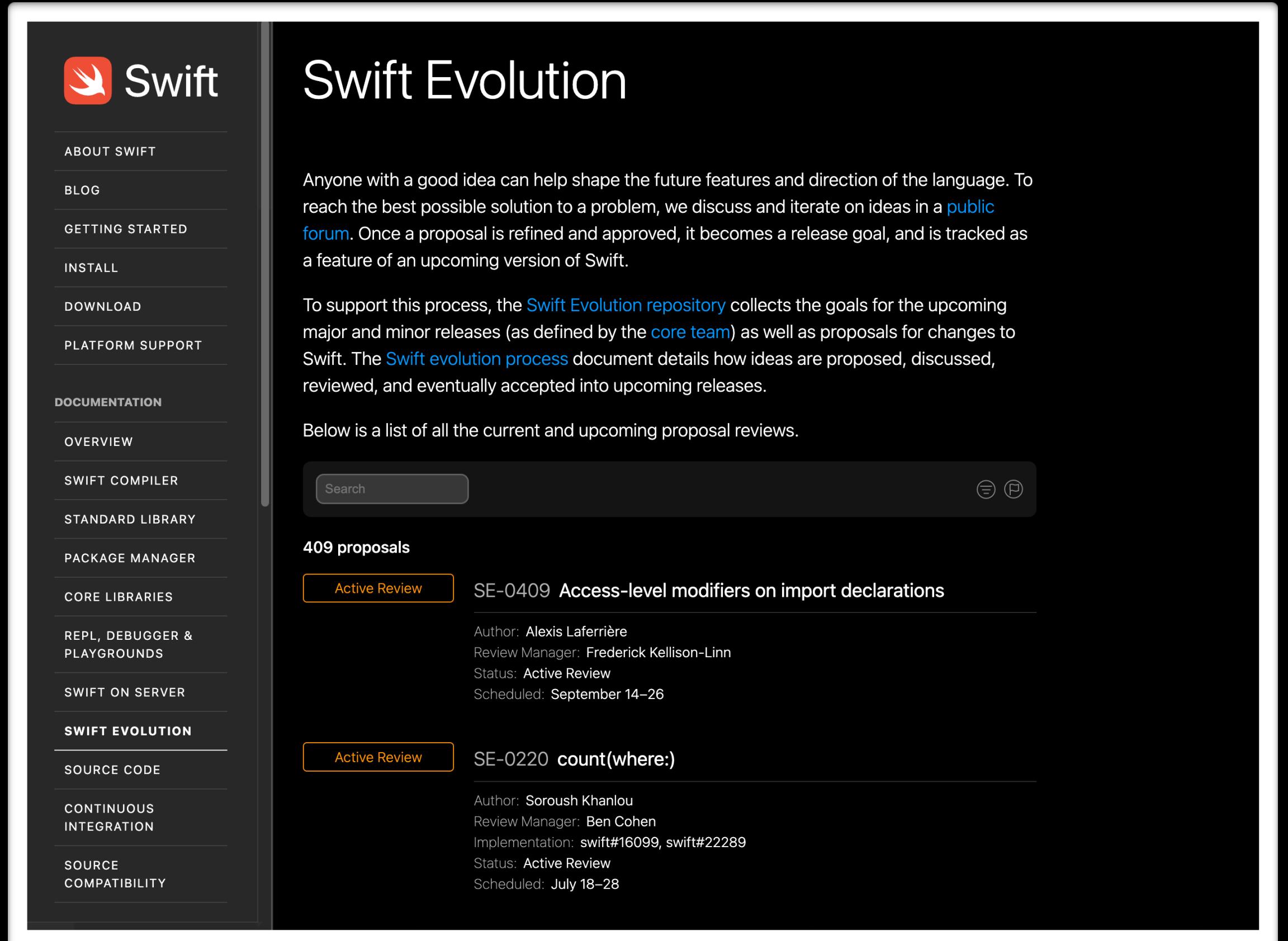


Swifting



Let'Swift 2023  
Deep Dive into the unknown

# 예시 프로젝트 및 코드 설명



The screenshot shows the Swift Evolution page. The left sidebar includes links for About Swift, Blog, Getting Started, Install, Download, Platform Support, Documentation (Overview, Swift Compiler, Standard Library, Package Manager), Core Libraries, REPL, Debugger & Playgrounds, Swift on Server, Swift Evolution (Active Review), Source Code, Continuous Integration, and Source Compatibility. The main content area features a title "Swift Evolution" and a paragraph about the evolution process. It then lists two active review proposals: "SE-0409 Access-level modifiers on import declarations" and "SE-0220 count(where:)". Each proposal entry includes the author's name, review manager, status, and scheduled date.

Proposal ID	Title	Author	Review Manager	Status	Scheduled Date
SE-0409	Access-level modifiers on import declarations	Alexis Laferrière	Frederick Kellison-Linn	Active Review	September 14–26
SE-0220	count(where:)	Soroush Khanlou	Ben Cohen	Active Review	July 18–28

<https://www.swift.org/swift-evolution/>



**Let'Swift 2023**  
Deep Dive into the unknown

# 예시 프로젝트 및 코드 설명

```
import ScadeKit

class BasePageAdapter<ViewModel: BaseViewModel>: SCDLatticePageAdapter {

    let viewModel: ViewModel

    init(viewModel: ViewModel) {
        self.viewModel = viewModel
    }

    override func load(_ path: String) {
        super.load(path)
        configUI()
        bindAction()
        bindState()
    }

    override func show(view: (SCDLatticeView)?, data: Any) {
        super.show(view: view, data: data)
        onAppear()
    }

    func configUI() {}
    func onAppear() {}
    func bindAction() {}
    func bindState() {}
}
```

BasePageAdapter.swift



# 예시 프로젝트 및 코드 설명



```
import ScadeKit
import Services

final class HomePageAdapter: BasePageAdapter<HomePageViewModel> {

    override func configUI() {
        mainList.elementProvider = SCDWidgetsElementProvider { (item: SwiftEvolutionEntity, template) in
            (template.getWidgetByName("statusLabel") as? SCDWidgetsLabel)?.text = item.status.transformedState
            (template.getWidgetByName("titleLabel") as? SCDWidgetsLabel)?.text = item.title
        }
    }

    override func bindAction() {
        viewModel.networkingAction {
            self.mainList.items = self.viewModel.swiftEvolution
        }
        mainList.onItemSelected.append(
            SCDWidgetsItemSelectedEventHandler { event in
                guard let item = event?.item as? SwiftEvolutionEntity else { return }
                self.viewModel.listItemSelect(item.link)
            }
        )
    }
}
```



# 예시 프로젝트 및 코드 설명

```
import Foundation
import Services

final class HomePageViewModel: BaseViewModel {

    var swiftEvolution: [SwiftEvolutionEntity] = []

    /// UseCase
    private let fetchSwiftEvolutionUseCase: SwiftEvolutionUseCase

    /// Initializer
    public init(
        fetchSwiftEvolutionUseCase: SwiftEvolutionUseCase
    ) {
        self.fetchSwiftEvolutionUseCase = fetchSwiftEvolutionUseCase
    }

    /// Swift Evolution API Networking Action
    func networkingAction(_ context: @escaping () -> Void) {
        Task { @MainActor in
            let item = try await fetchSwiftEvolutionUseCase.execute()
            swiftEvolution = item.reversed()
            context()
        }
    }

    /// Item Select Action
    func listItemSelect(_ url: String) {
        Navigation.go(.detail(url), transition: .fromTop)
    }
}
```

home.page.viewModel.swift



**Let'Swift 2023**  
Deep Dive into the unknown

# 예시 프로젝트 및 코드 설명

```
extension NetworkImpl {
    public func request<T: Decodable>(_ api: API, dto: T.Type) async throws -> T {
        let requestData = try await performRequest(context: api)
        return try JSONDecoder().decode(dto.self, from: requestData)
    }

    public func requestArray<T: Decodable>(_ api: API, dto: T.Type) async throws -> [T] {
        let requestData = try await performRequest(context: api)
        return try JSONDecoder().decode([T].self, from: requestData)
    }

    public func performRequest(context api: API) async throws -> Data {
        do {
            let request = try api.toRequest()
            let (data, response) = try await requestable.data(for: request)
            guard let httpResponse = response as? HTTPURLResponse else {
                throw NetworkingError(reason: .noResponse)
            }
            guard api.validationCode == httpResponse.statusCode else {
                throw StatusError(reason: StatusErrorReason(statusCode: httpResponse.statusCode))
            }
            return data
        } catch {
            guard let errorDict = api.errorDict,
                  let statusError = error as? StatusError,
                  let apiError = errorDict[statusError.reason.statusCode] else {
                throw error
            }
            throw apiError
        }
    }
}
```

Network.swift



# 예시 프로젝트 및 코드 설명

```
struct SwiftEvolutionResponseDTO: Decodable {
    let authors: [Authors]
    let id: String
    let link: String
    let reviewManager: ReviewManager?
    let sha: String
    let status: Status
    let summary: String
    let title: String
    let trackingBugs: [TrackingBugs]?
    let warnings: [Warnings]?

    struct Authors: Decodable {
        let link: String
        let name: String
    }

    struct ReviewManager: Decodable {
        let link: String
        let name: String
    }

    struct Status: Decodable {
        let state: String
        let version: String?
    }

    struct TrackingBugs: Decodable {
        let assignee: String
        let id: String
        let link: String
        let radar: String
        let resolution: String
        let status: String
        let title: String
        let updated: String
    }

    struct Warnings: Decodable {
        let kind: String
        let message: String
        let stage: String
    }
}
```

```
extension SwiftEvolutionResponseDTO {
    func toDomain() -> SwiftEvolutionEntity {
        return SwiftEvolutionEntity(
            authors: authors.map { $0.toDomain() },
            id: id,
            link: link,
            sha: sha,
            status: status.toDomain(),
            summary: summary,
            title: title
        )
    }
}

extension SwiftEvolutionResponseDTO.Authors {
    func toDomain() -> AuthorsEntity {
        AuthorsEntity(
            link: link,
            name: name
        )
    }
}

extension SwiftEvolutionResponseDTO.Status {
    func toDomain() -> StatusEntity {
        StatusEntity(
            state: state,
            version: version
        )
    }
}
```

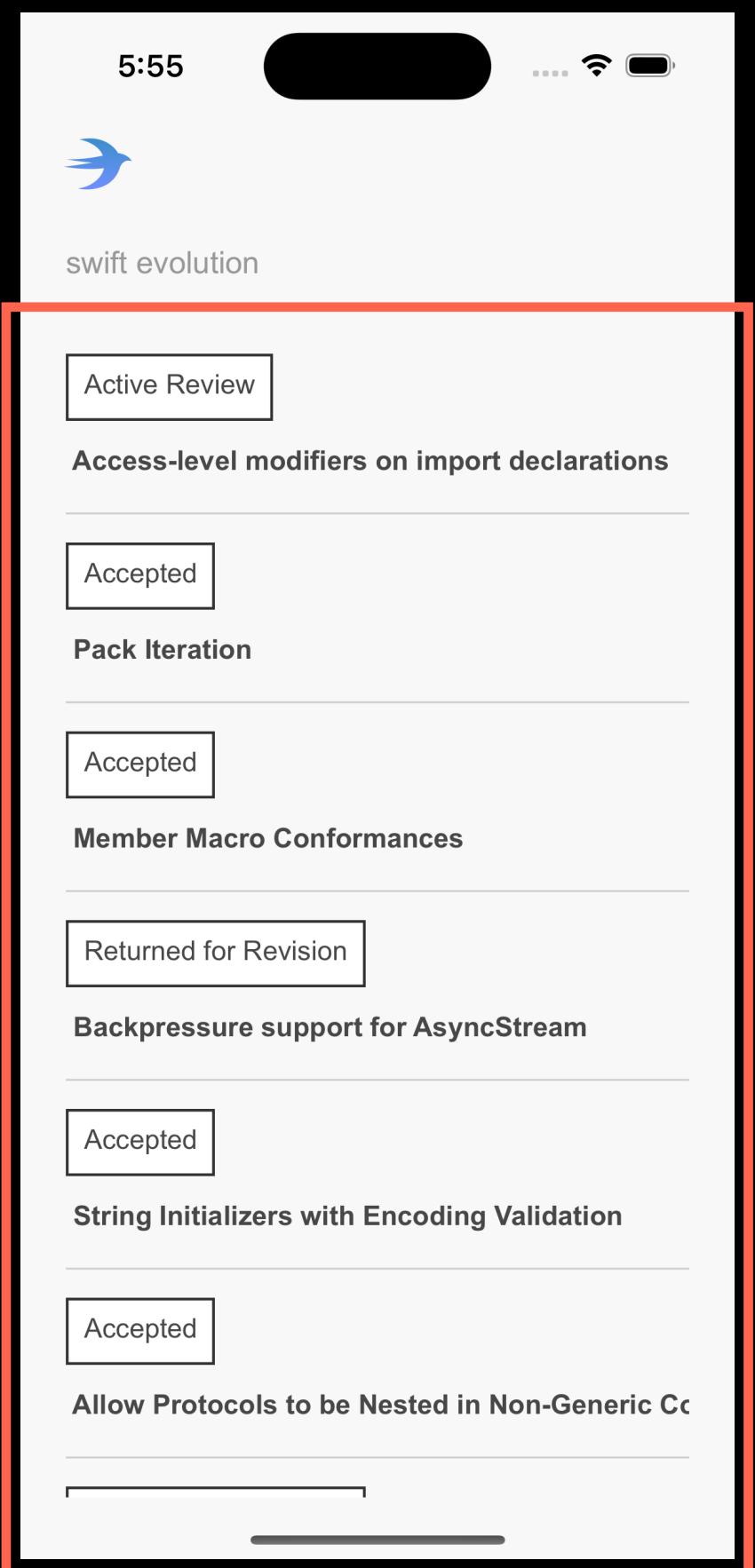


# 예시 프로젝트 및 코드 설명

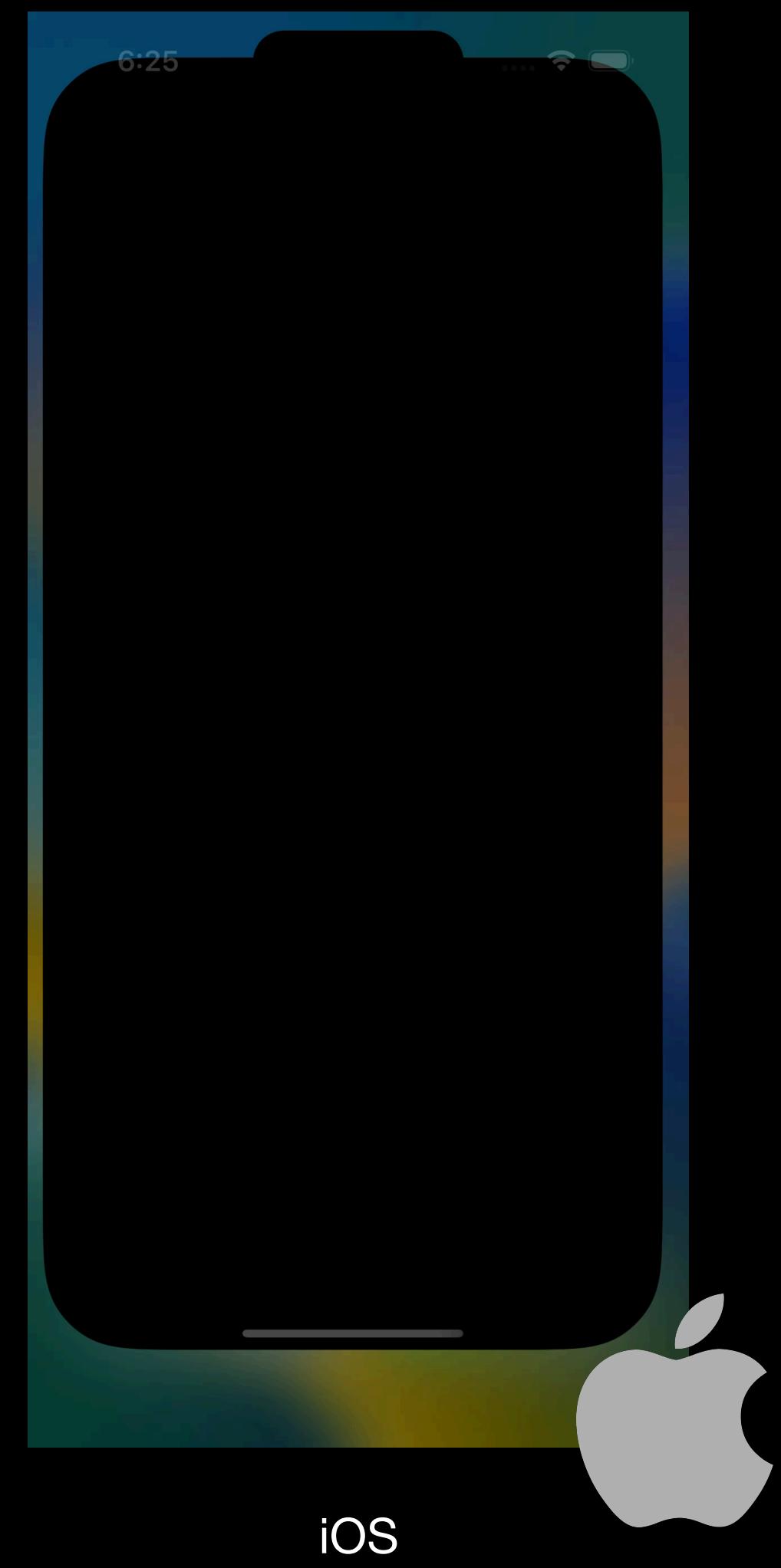
```
import Foundation
import ScadeKit

public final class SwiftEvolutionEntity: EObject {
    public let authors: [AuthorsEntity]
    public let id: String
    public let link: String
    public let sha: String
    public let status: StatusEntity
    public let summary: String
    public let title: String

    public init(
        authors: [AuthorsEntity],
        id: String,
        link: String,
        sha: String,
        status: StatusEntity,
        summary: String,
        title: String
    ) {
        self.authors = authors
        self.id = id
        self.link = link
        self.sha = sha
        self.status = status
        self.summary = summary
        self.title = title
    }
}
```



# 예시 프로젝트 및 코드 설명



**Let'Swift 2023**  
Deep Dive into the unknown

# 장단점 설명



**Let'Swift 2023**  
Deep Dive into the unknown

# 장단점 설명

## 장점

- 익숙한 UI의 IDE
- iOS, Android 동시개발 가능
- Swift 언어를 채택하기 때문에, 언어 학습의 리소스 소비 X
- 꾸준한 업데이트 (2017~현재)
- Storyboard 와 비슷한 Page 기능을 지원
- 공식 Slack 커뮤니티로 Scade 에 질문 가능

## 단점

- IDE에 부족한 점이 보임 (page 부분 또는 자잘한 이슈들)
- 지원되는 UI Controls의 갯수 부족
- 현재도 베타 버전..
- Launch Screen 지원 안함
- 오픈소스 문서들이 친절하다고 느껴지지는 않음 (사용법만 올려놓은 느낌)



마무리



Let'Swift 2023  
Deep Dive into the unknown

# 마무리



## Swift

# 마무리



Swift



Apple Platform

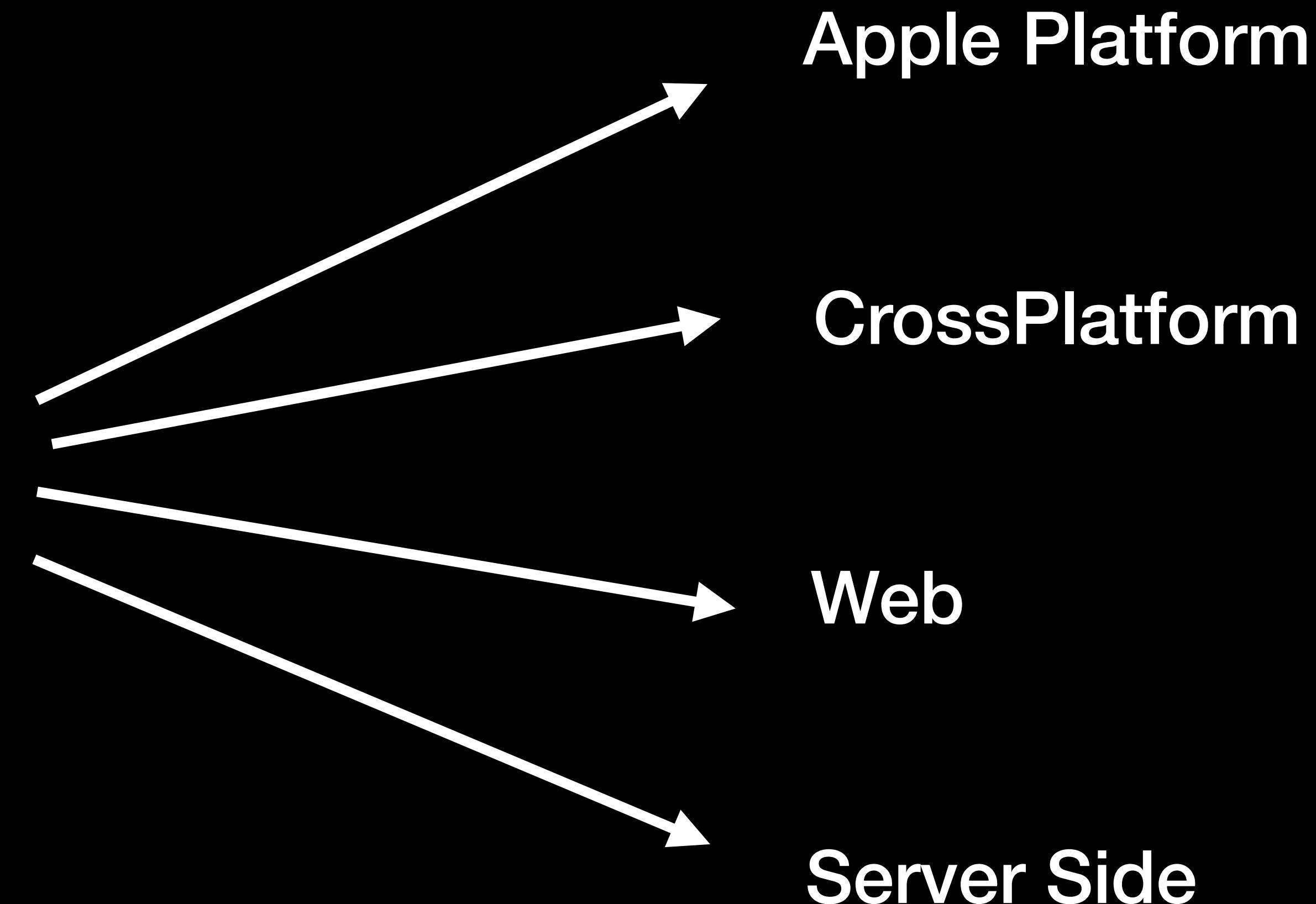


Let'Swift 2023  
Deep Dive into the unknown

# 마무리



Swift

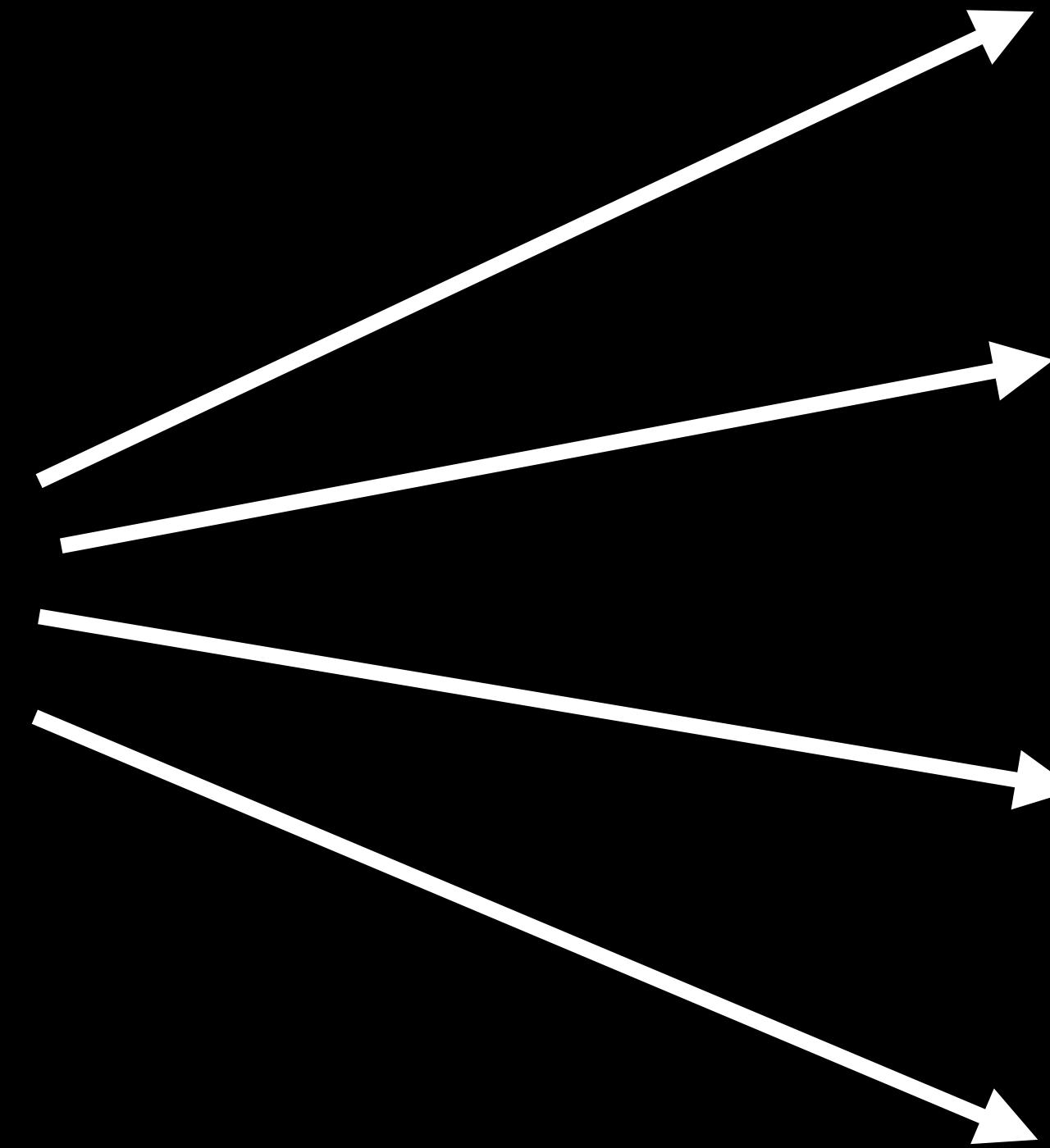


**Let'Swift 2023**  
Deep Dive into the unknown

# 마무리



Swift



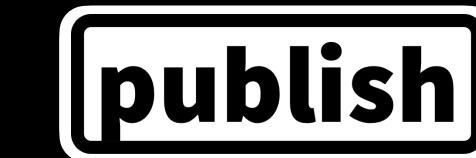
Apple Platform



CrossPlatform



Web



Swift WASM

Server Side



Let'Swift 2023  
Deep Dive into the unknown

# 마무리

개인적으로  는..



**Let'Swift 2023**  
Deep Dive into the unknown

# 마무리



Swift



CrossPlatform



Let'Swift 2023  
Deep Dive into the unknown

# 마무리

Component	Support
Swift	Yes. Swift 5.x on both Android and iOS
Swift Foundation	Yes. Works on both Android and iOS
Swift Dispatch	Yes. Works on both Android and iOS
iOS frameworks - kit support	Yes. Each and every kit works on iOS. Your SCADE iOS app is never constrained and can use the full iOS SDK, ie. UIKit, PassKit ... without limitations.
Swift Libraries	Yes. All your favourite libraries that use pure Swift and Swift Foundation and Switch Dispatch work <b>on iOS and Android</b> . Most of your favourite libraries just work, i.e. SwiftData, SwiftMoment, SQLite, FileKit,....
SwiftUI	SwiftUI is a central concept and very important part of the future of SCADE. Currently, you can use the optional command directive to use SwiftUI in the iOS part of your SCADE app. For using SwiftUI to Android and iOS as a cross platform solution, we are working on an elegant solution which will be part of a later major release.
SPM	Yes. Works nicely.



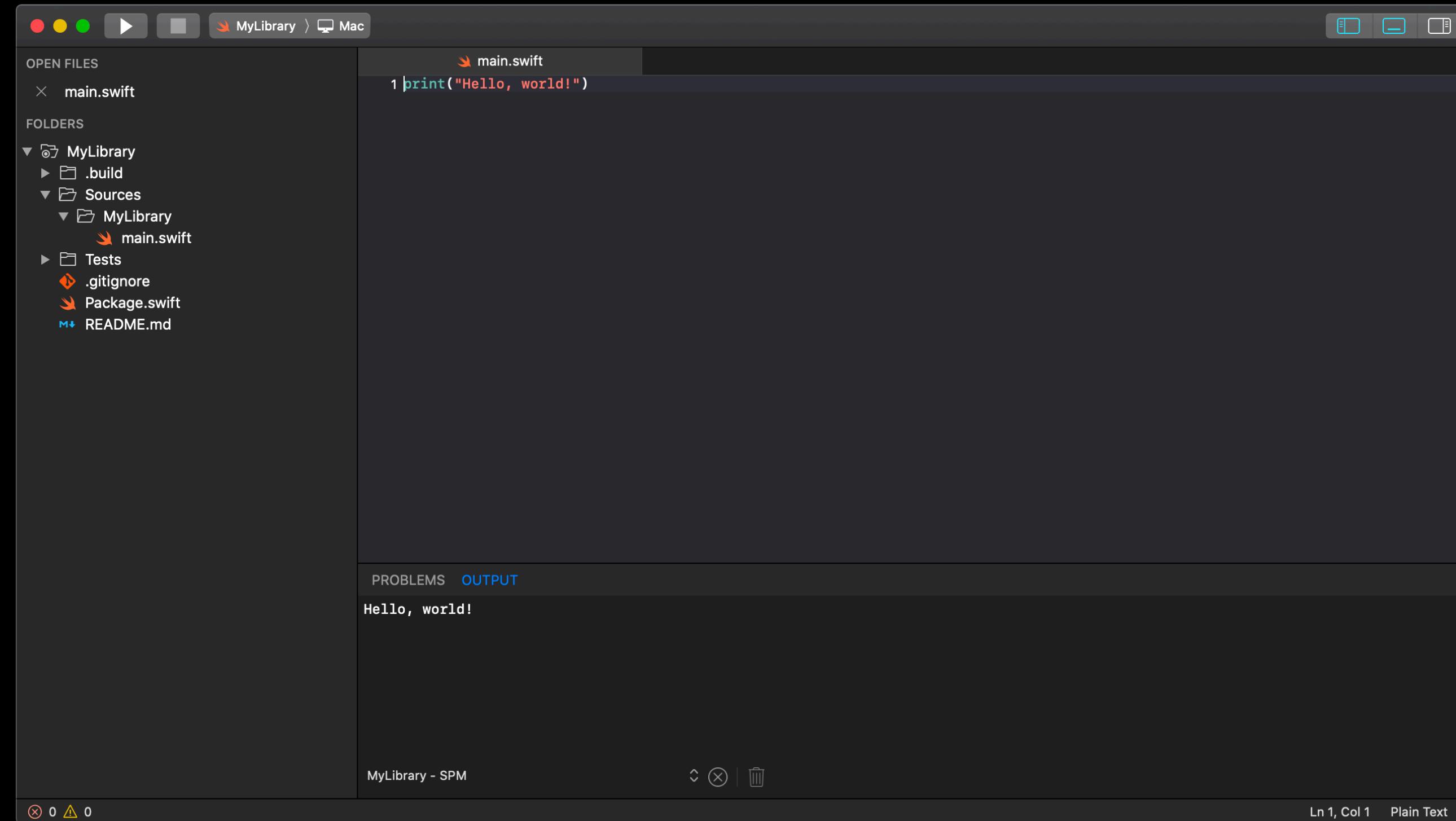
추후 SwiftUI 지원을 중요하게 생각함

<https://docs.scade.io/docs/how-scade-works>



**Let'Swift 2023**  
Deep Dive into the unknown

# 마무리



```
1 print("Hello, world!")
```

OPEN FILES  
x main.swift

FOLDERS  
▼ MyLibrary  
  ► .build  
  ▼ Sources  
    ► MyLibrary  
      main.swift  
  ► Tests  
  ► .gitignore  
  ► Package.swift  
  ► README.md

PROBLEMS OUTPUT  
Hello, world!

MyLibrary - SPM

Ln 1, Col 1 Plain Text

0 0 0



<https://github.com/scade-platform/Nimble>

# Thank You



**Let'Swift 2023**  
Deep Dive into the unknown



# Let'Swift 2023

Deep Dive into the unknown