

iOS Seminar 0

Wafflestudio 2025



세미나 Overview

세미나장 소개

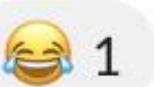
- 최유림
- Wafflestudio 19.5기 iOS
 - SNUTT iOS & PM
- 컴퓨터공학부 19 / 시각디자인



iOS 개발의 장단점

- 장점
 - 예쁘다! (끝)
 - 기본 UI Component만 잘 짜맞춰도 낫배드
 - iOS를 위한 코드 하나만 작성해도 iPad, Macbook에서 설치 가능
- 단점
 - 매년 WWDC가 진행됨에 따라 새롭게 공부할 것들이...
 - YouTube (2024년 2월 개설)

4월 3일 오후 10:06
수치보기귀차나

최유림 4월 3일 오후 10:07
 자꾸 그러면 안드 더 못생겨져
 1 

4월 3일 오후 10:07
진짜못생긴거보여줘?
 1 

세미나 일정

- 0회: 9월 6일 (토)
- 1회: 9월 20일 (토)
- 2회: 10월 18일 (토)
- 3회: 11월 1일 (토)
- 4회: 11월 29일 (토)

과제 관련

- **과제 마감 : 다음 세미나 당일 오전 8시까지**
 - ex) 과제 1 마감: 10월 18일 토 오전 8시
- Grace Day
 - 모각코(2인 이상) 참여: 2회 당 1일 추가 → 최대 5일
 - 각 과제의 Bonus 스펙을 구현: 2개 당 1일 추가 → 제한 없음
- 모든 Grace Day는 한 과제에 몰아서 사용하셔도 됩니다!

통과 기준

- 전 세미나 대면 출석
- 모든 과제 제출&통과
- 모각코 2회 이상 참여 (와플 전체 / iOS 단독 무관)

Bonus 스펙 2개 구현 완료

 Grace Day 1일 추가입니다

- 아이디를 입력하는 UITextField에 영어를 입력하면 자동으로 첫 글자가 대문자가 되는 현상을 해결합니다
- 마지막으로 로그인할 때 사용했던 아이디는 다음으로 앱을 켤 때 자동으로 아이디 UITextField에 채워져 있도록 합니다



peng-u-0807 commented on Nov 7, 2024

...

Grace Day 3일(기본 2일+보너스 스펙 1일)에서 과제2 지각 제출(11월 3일 오전)로 하루 차감되어 2일 남았습니다.
따로 계산하신 값과 다르다면 알려주세요:)



무엇을 다루나요?

- UIKit, SwiftUI를 이용한 UI 구현
- Alamofire를 이용한 네트워크 통신
- Concurrency
- Design Pattern
- 그 외 ...

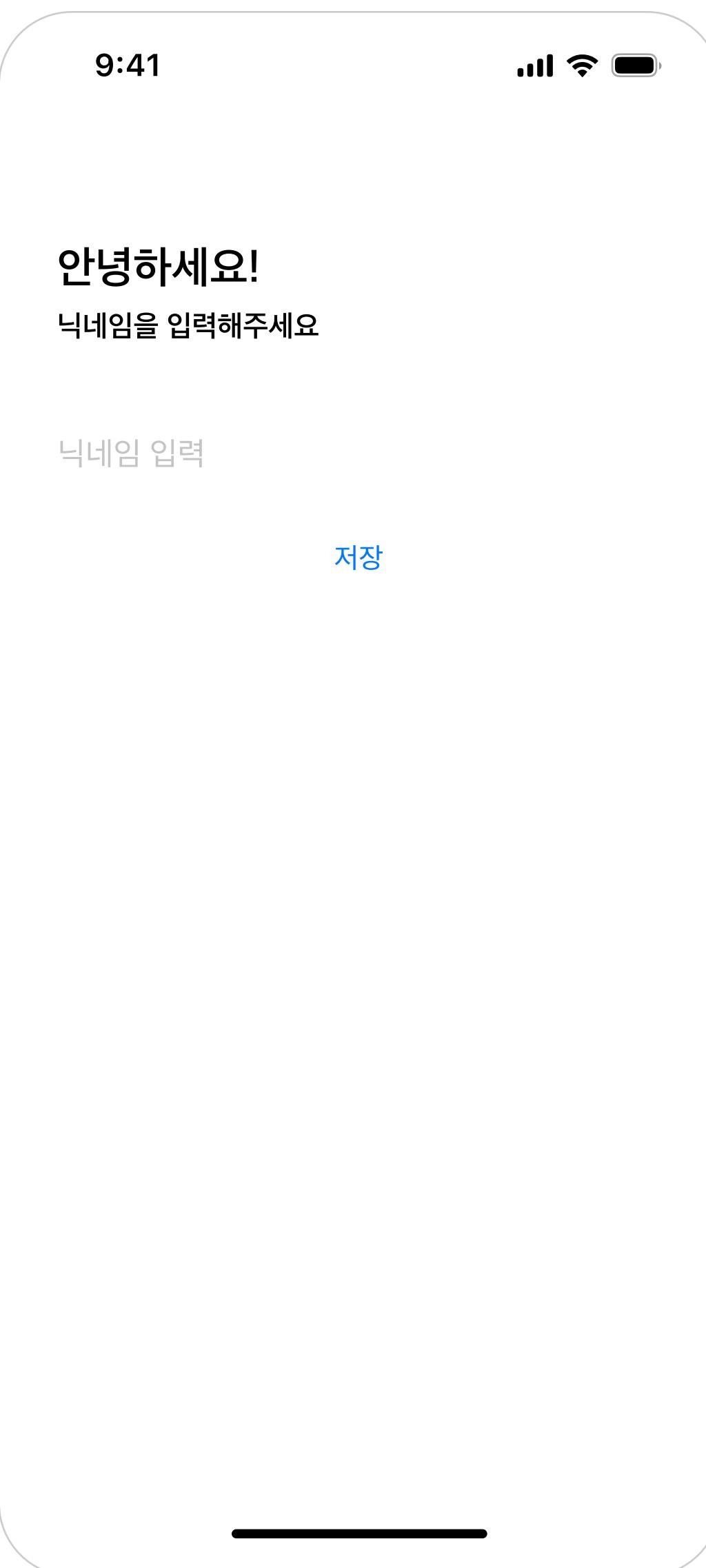
어떻게 공부하면 좋을까요?

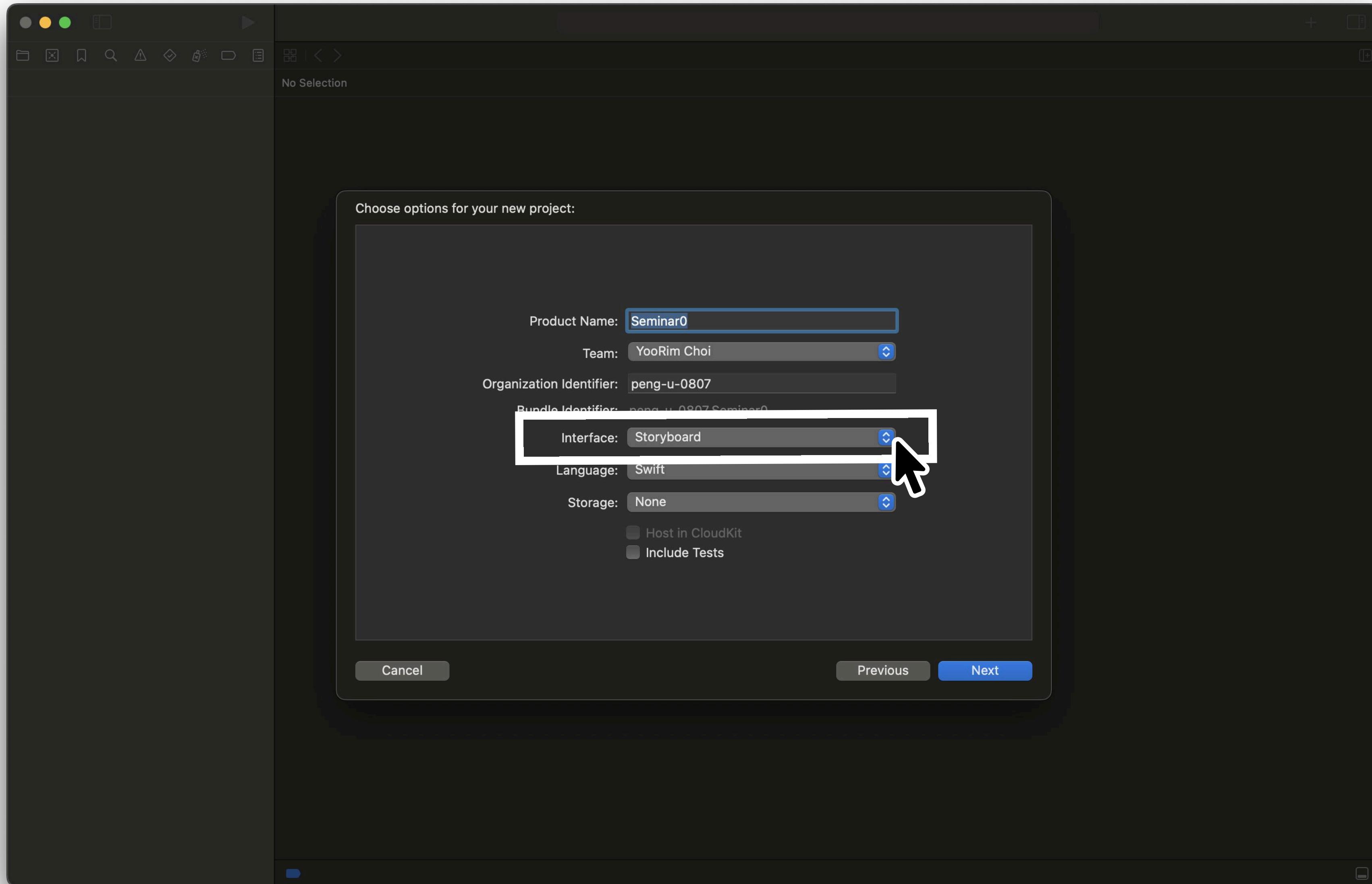
- [UIKit 공식 문서](#)
- [SwiftUI 공식 문서](#)
- (그 외 수많은 구글링)
 - 단, 작성 일자에 주의
 - 갑자기 Objective-C를 만날 수도...

Storyboard

iOS 개발에서 UI를 구현하는 방법

- UIKit
 - 명령형 프로그래밍
 - Storyboard
 - No Storyboard (Programmatic UI)
- SwiftUI
 - 선언형 프로그래밍
 - 세미나3에서 다룰 예정



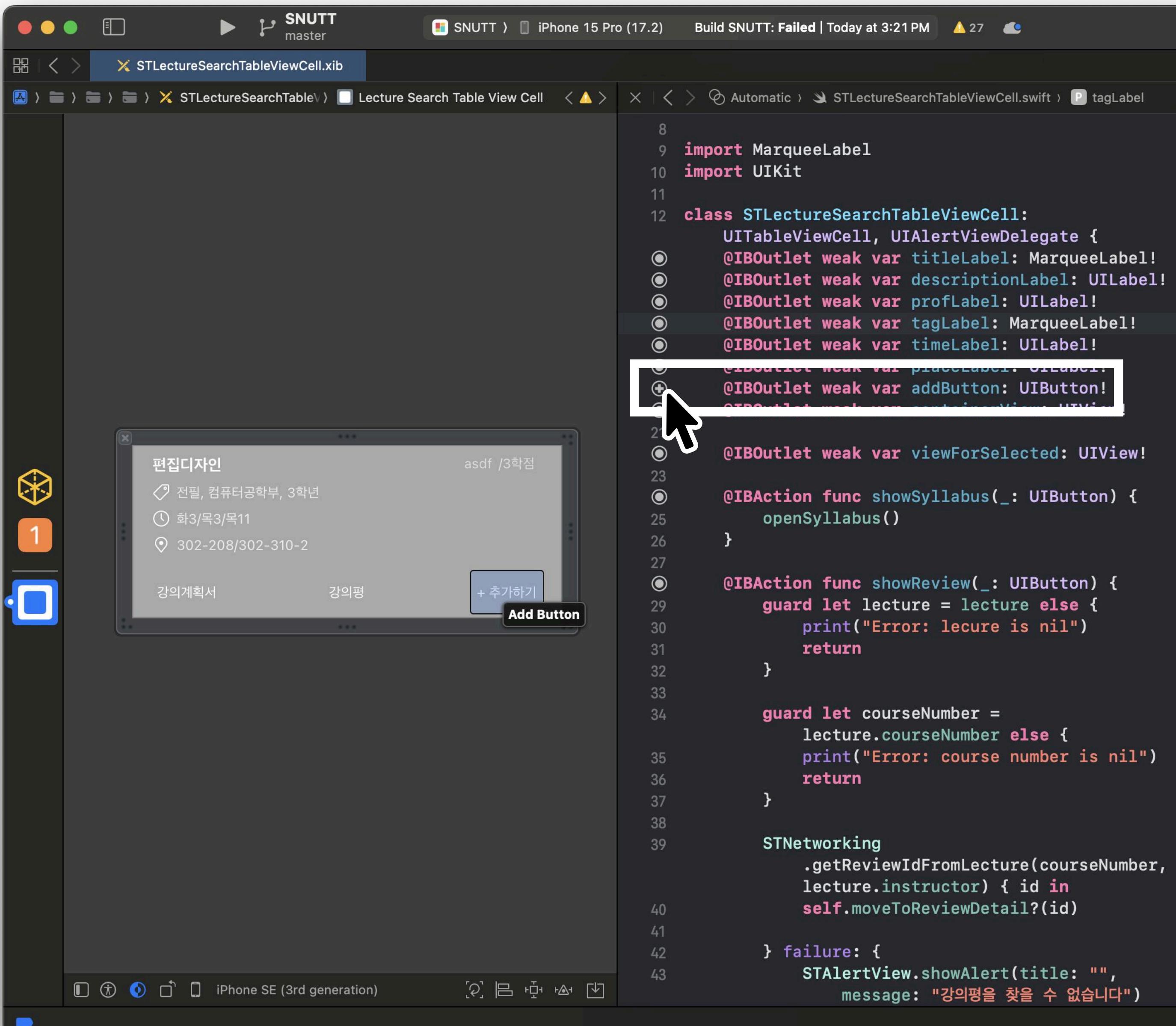
 Storyboard

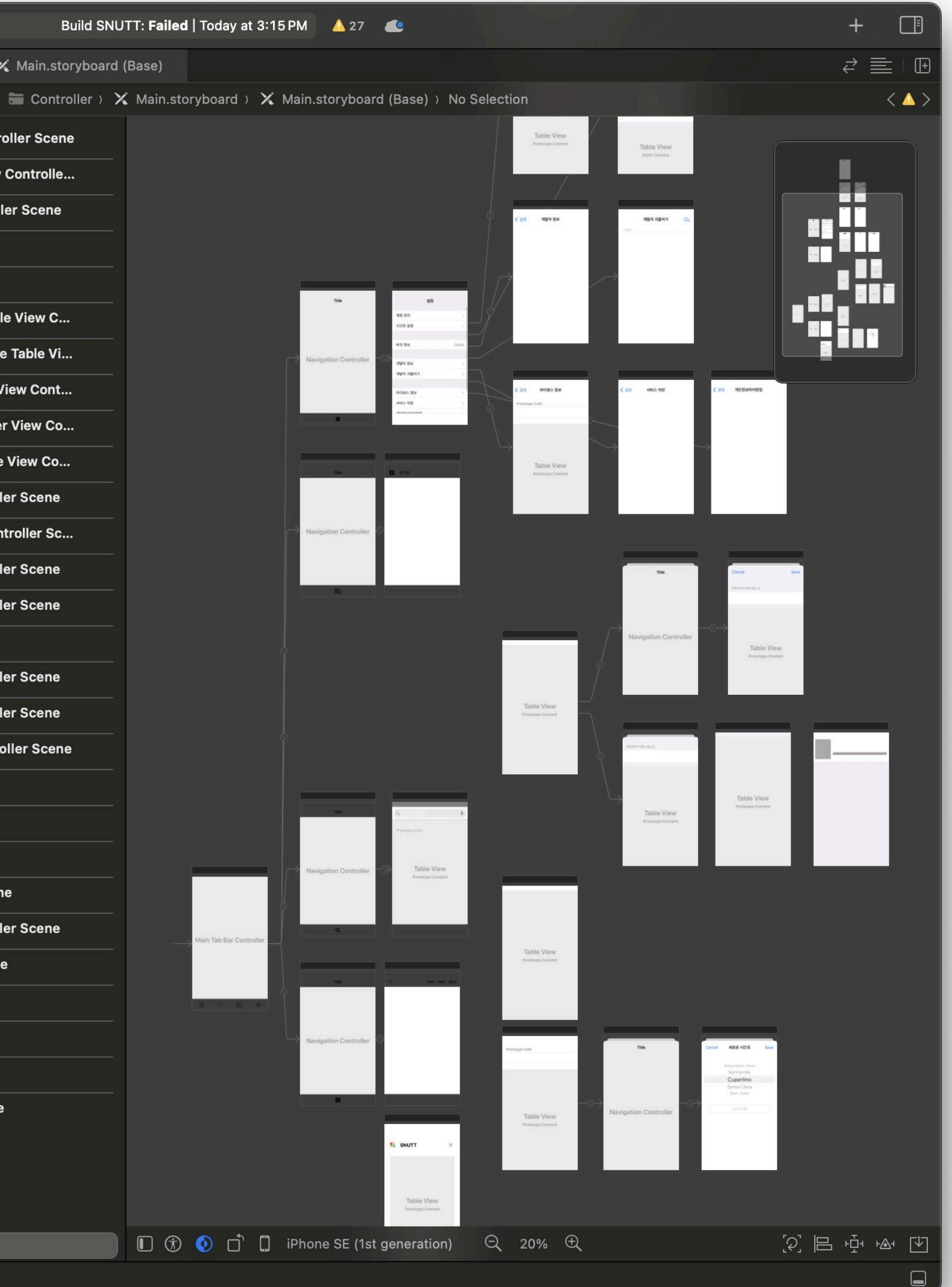
Storyboard

Storyboard의 장점

- 전체 UIView를 한 눈에 확인
- Drag & Drop → UIView 추가
- Hover Action → 연결된 UIView 확인

→ 직관적인 인터페이스



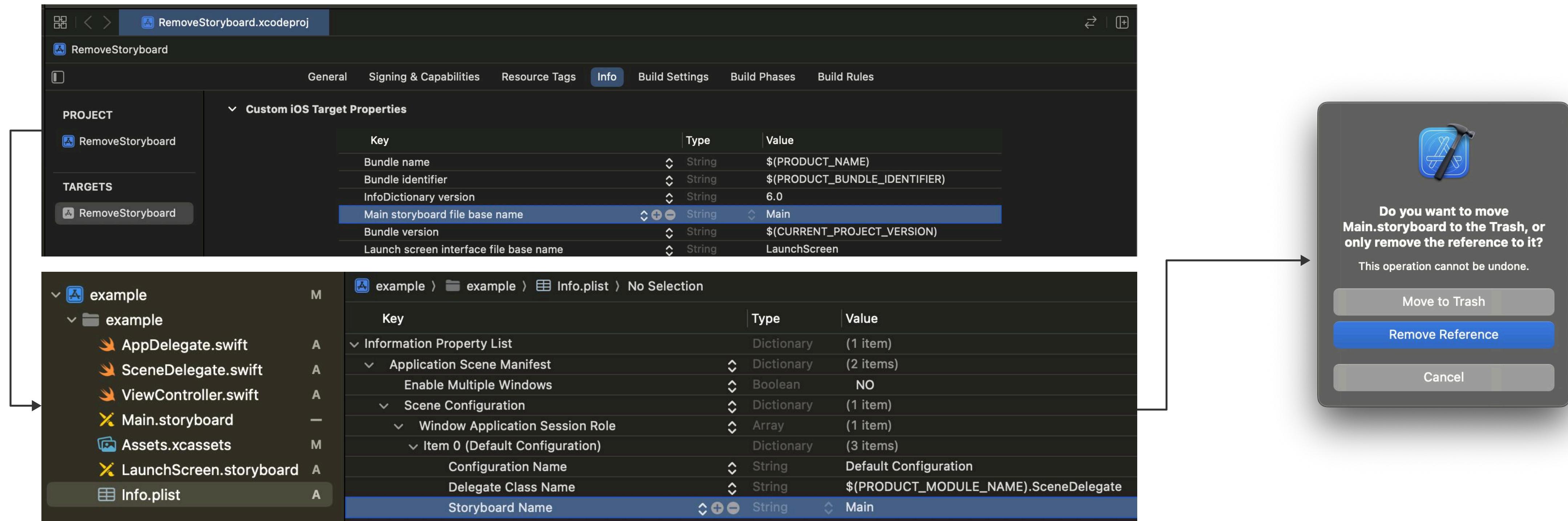


그러나..

- 프로젝트의 규모가 커질수록 관리가 어렵다
- 수정 사항 반영 및 실행에 오랜 시간이 걸린다

→ No Storyboard

(1) Main.storyboard 삭제



1. TARGETS > {ProjectName} > Info > “Main storyboard file base name” > Backspace
2. Info.plist > ... > “Storyboard Name” > Backspace
3. Main.storyboard 클릭 > Backspace > “Move to Trash”

(2) SceneDelegate 변경

```
10 class SceneDelegate: UIResponder, UIWindowSceneDelegate {  
11  
12     var window: UIWindow?  
13  
14  
15     func scene(_ scene: UIScene, willConnectTo session: UISceneSession, options connectionOptions:  
16                 UIScene.ConnectionOptions) {  
17         // Use this method to optionally configure and attach the UIWindow `window` to the provided  
18         // UIWindowScene `scene`.  
19         // If using a storyboard, the `window` property will automatically be initialized and attached to  
20         // the scene.  
21         // This delegate does not imply the connecting scene or session are new (see  
22             // `application:configurationForConnectingSceneSession` instead).  
23         guard let _ = (scene as? UIWindowScene) else { return }  
24     }  
25 }
```

```
10 class SceneDelegate: UIResponder, UIWindowSceneDelegate {  
11  
12     var window: UIWindow?  
13  
14  
15     func scene(_ scene: UIScene, willConnectTo session: UISceneSession, options connectionOptions:  
16                 UIScene.ConnectionOptions) {  
17         guard let windowScene = (scene as? UIWindowScene) else { return }  
18         window = UIWindow(frame: windowScene.coordinateSpace.bounds)  
19         window?.windowScene = windowScene  
20         window?.rootViewController = UINavigationController(rootViewController: ViewController())  
21         window?.makeKeyAndVisible()  
22     }  
23 }
```

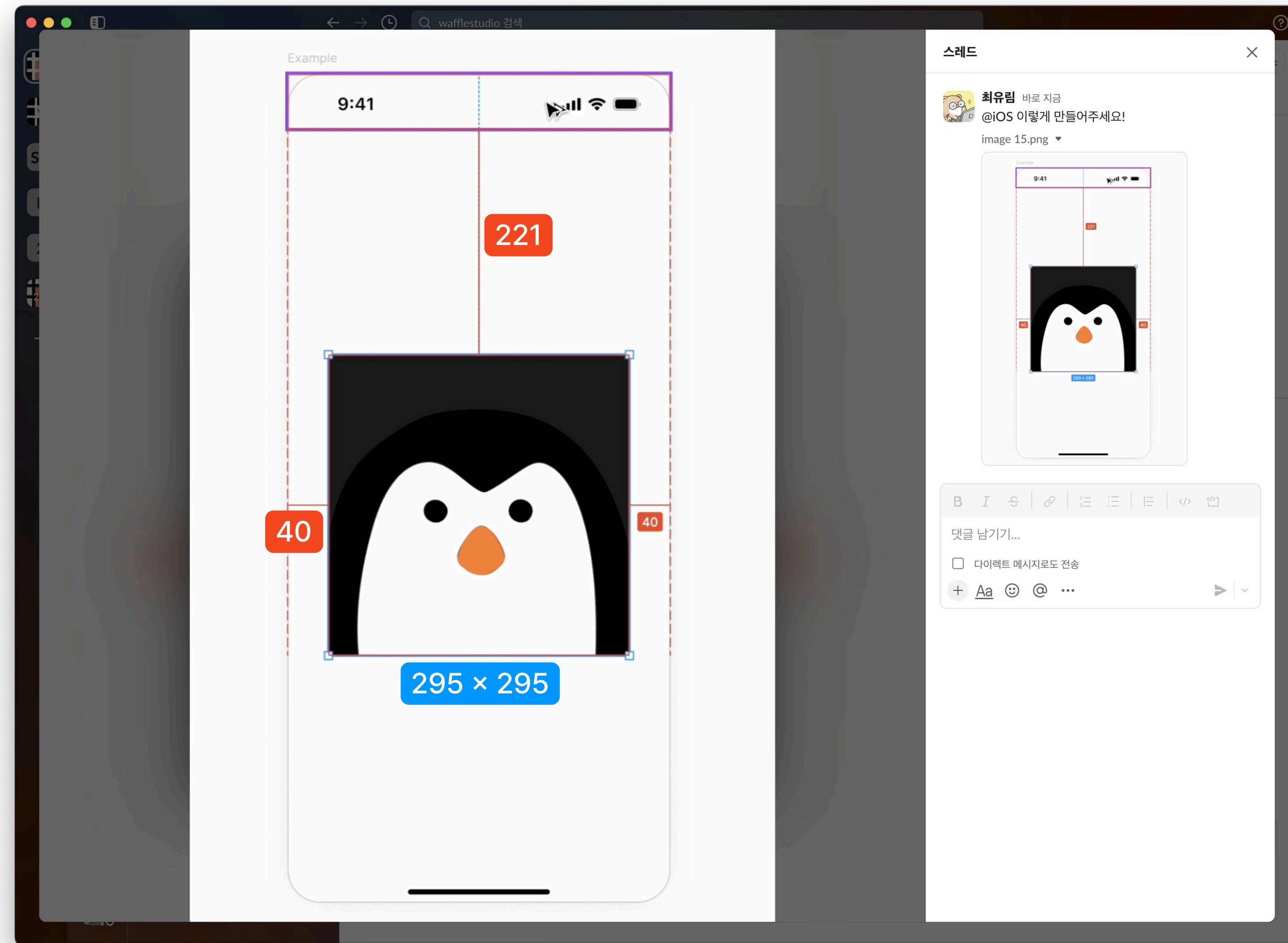
- SceneDelegate란?
- SceneDelegate의 다른 메서드들이 궁금하다면... [참고 링크](#)

Auto Layout

Auto Layout

- 다른 View를 기준으로 일정한 Constraint를 주어서 View를 상대적으로 그리는 방법
 - Auto + Layout
 - Auto Layout dynamically calculates the size and position of all the views in your view hierarchy, based on constraints placed on those views.
- 만약 Auto Layout을 적용하지 않는다면?
 - iPhone Mini vs iPad 12.9

Auto Layout

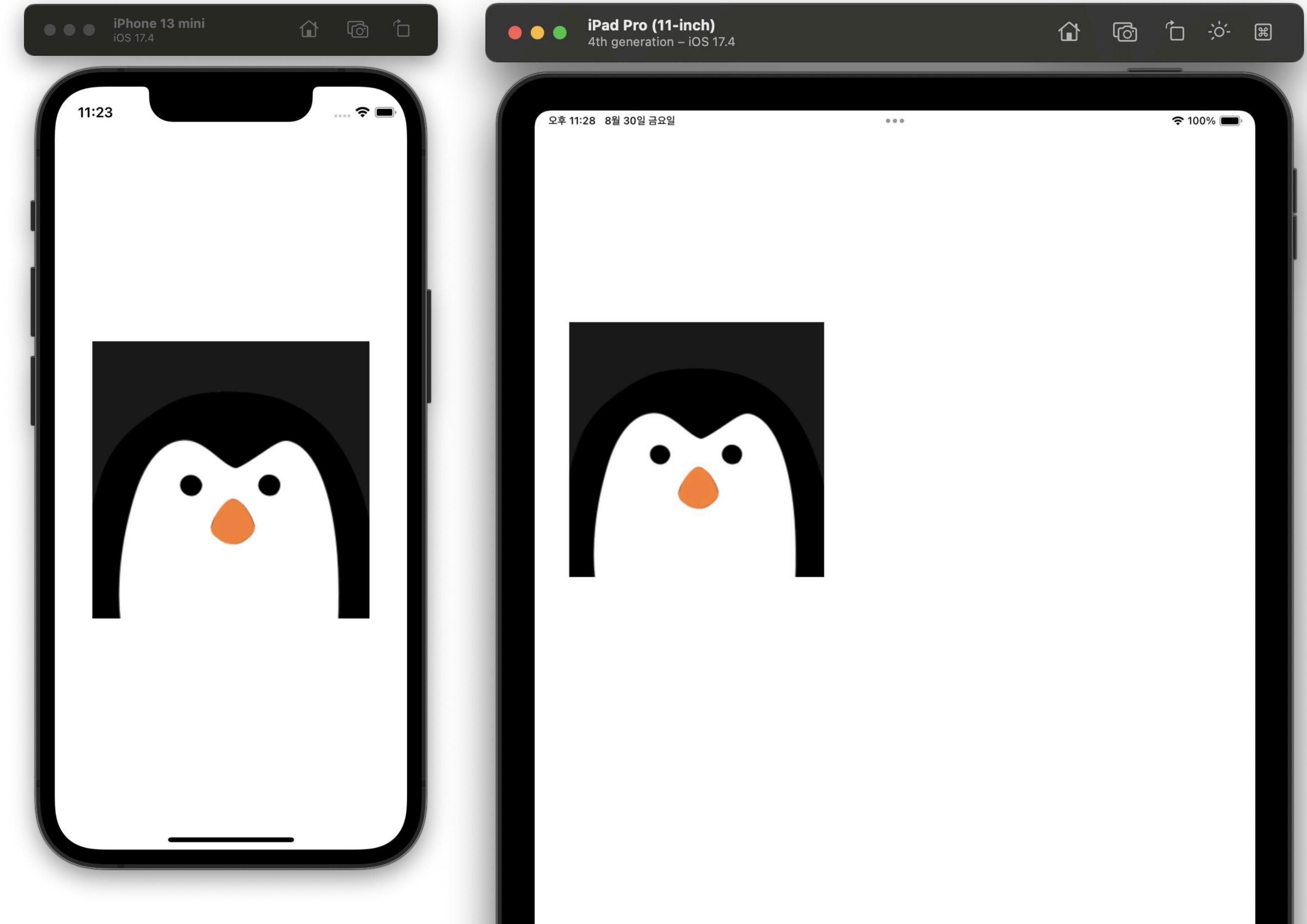


```
8 import UIKit  
9  
10 class ViewController: UIViewController {  
11  
12     private lazy var penguinImageView: UIImageView = {  
13         let image = UIImage(named: "Penguin")  
14         return .init(image: image)  
15     }()  
16  
17     override func viewDidLoad() {  
18         super.viewDidLoad()  
19         view.backgroundColor = .white  
20         view.addSubview(penguinImageView)  
21         penguinImageView.frame = .init(x: 40, y: 221, width: 295, height: 295)  
22     }  
23 }
```

PenguinViewController.swift

Auto Layout을 적용하지 않는다면?

- 이런 결과를 의도하진 않았을 것
(대부분의 경우)

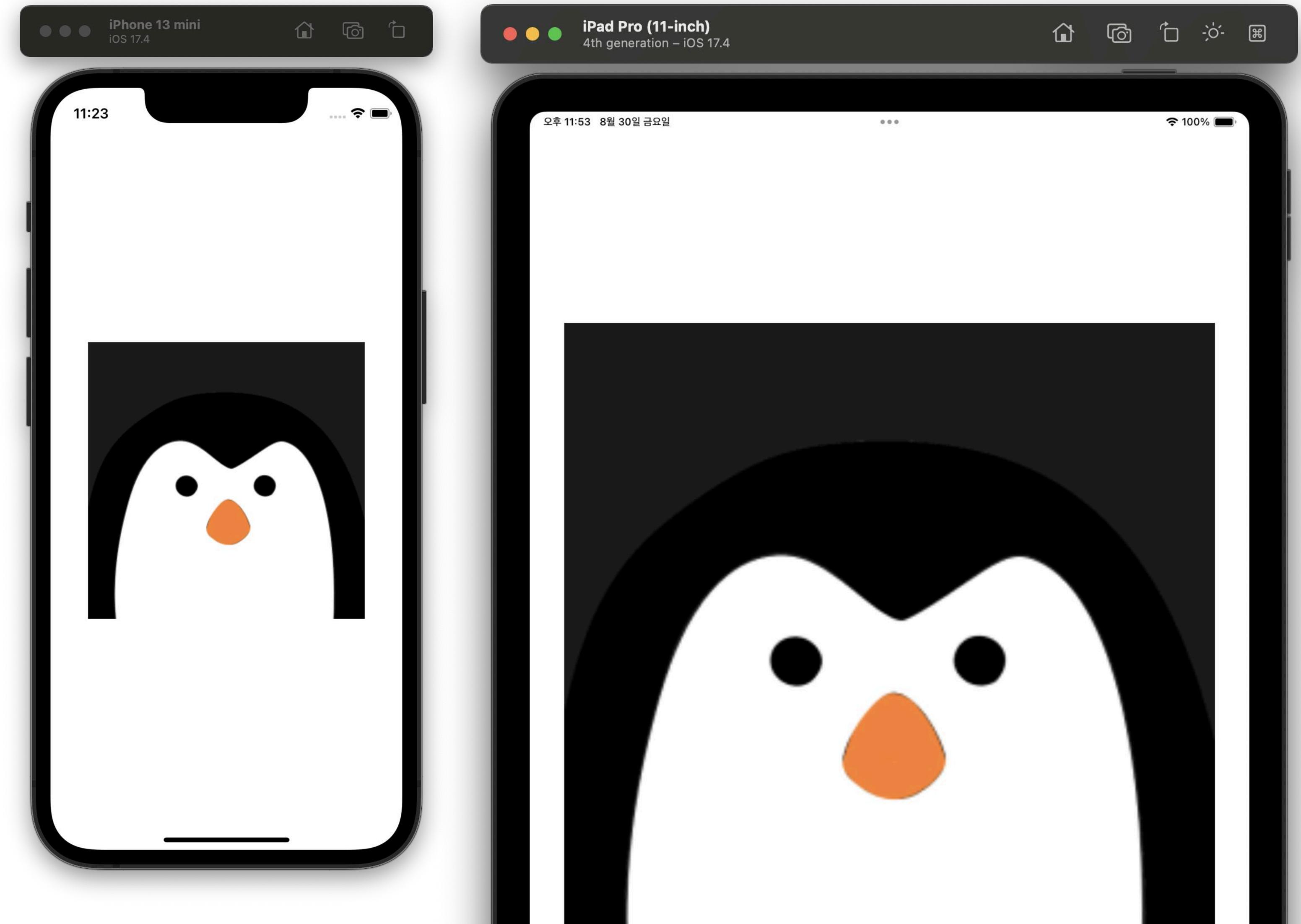


```
8 import UIKit
9
10 class ViewController: UIViewController {
11
12     private lazy var penguinImageView: UIImageView = {
13         let image = UIImage(named: "Penguin")
14         return .init(image: image)
15     }()
16
17     override func viewDidLoad() {
18         super.viewDidLoad()
19         view.backgroundColor = .white
20         view.addSubview(penguinImageView)
21         penguinImageView.translatesAutoresizingMaskIntoConstraints = false
22         NSLayoutConstraint.activate([
23             penguinImageView.widthAnchor.constraint(equalTo: penguinImageView.heightAnchor),
24             penguinImageView.topAnchor.constraint(equalTo: view.safeAreaLayoutGuide.topAnchor, constant: 221),
25             penguinImageView.leadingAnchor.constraint(equalTo: view.safeAreaLayoutGuide.leadingAnchor, constant: 40),
26             penguinImageView.trailingAnchor.constraint(equalTo: view.safeAreaLayoutGuide.trailingAnchor, constant: -40)
27         ])
28     }
29 }
```

AutoLayoutPenguinViewController.swift

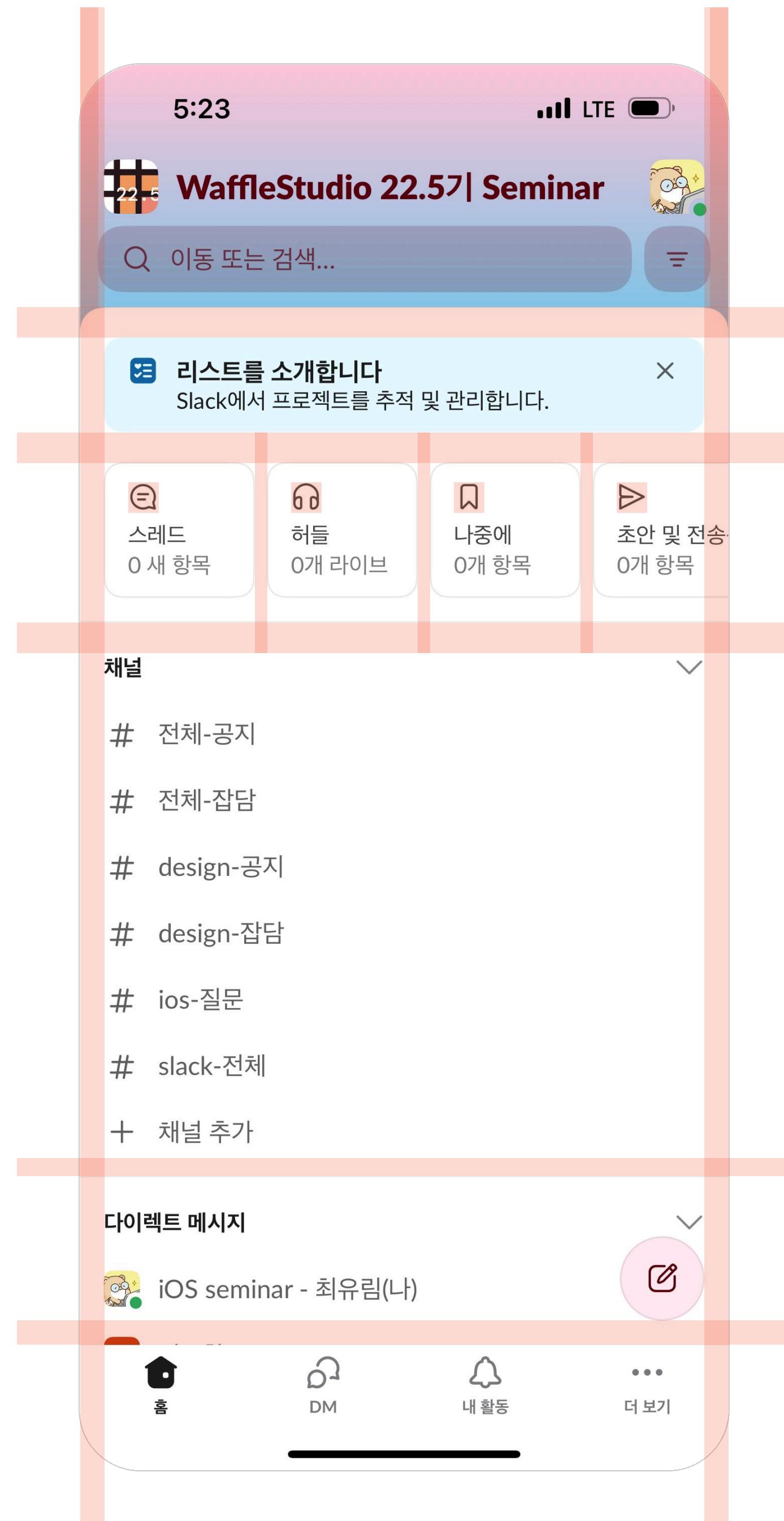
Auto Layout 적용 후

- 다양한 크기의 기기에서
동일한 모습의 View 유지



Auto Layout을 어떻게 잡아야 할까?

- 기기의 크기가 달라질 때, 변하는 부분과 변하지 않는 부분 생각
 - 예시) 오른쪽 사진에서 붉은색 직사각형: 어느 기기든 동일
 - 워크스페이스 이름, 검색창 등: 기기에 따라 가로의 길이 변함
- 어느 기기에서든 변하지 않는 부분을 constraint로 설정
 - 상위뷰(Superview)의 크기가 바뀌어도
변하지 않는 값을 제약으로 걸고(constraint)
나머지 레이아웃은 자동으로 잡아달라는 것



Constraints

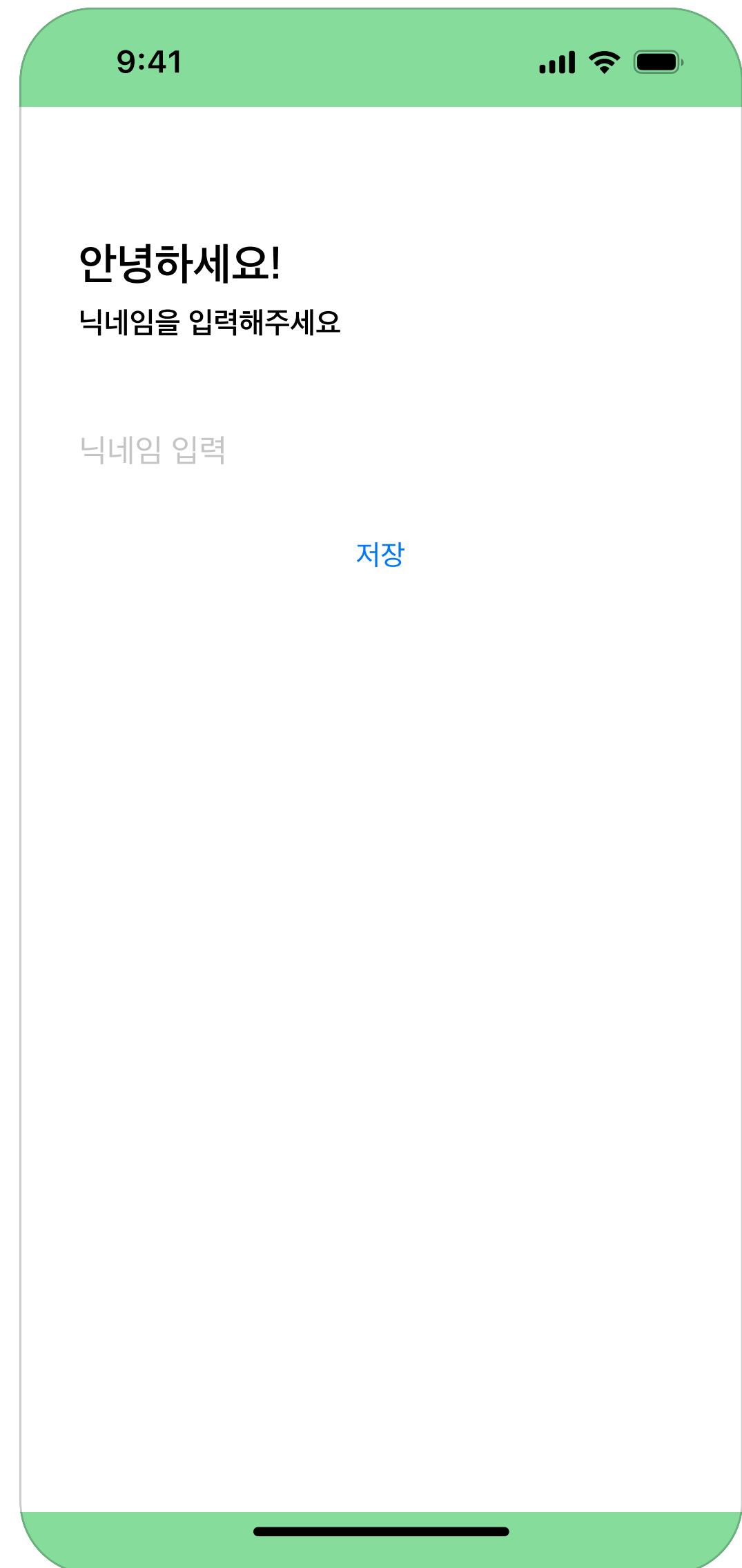
- topAnchor / bottomAnchor / leadingAnchor / trailingAnchor / leftAnchor / rightAnchor
- centerXAnchor / centerYAnchor
- heightAnchor / widthAnchor

leadingAnchor & trailingAnchor vs leftAnchor & rightAnchor

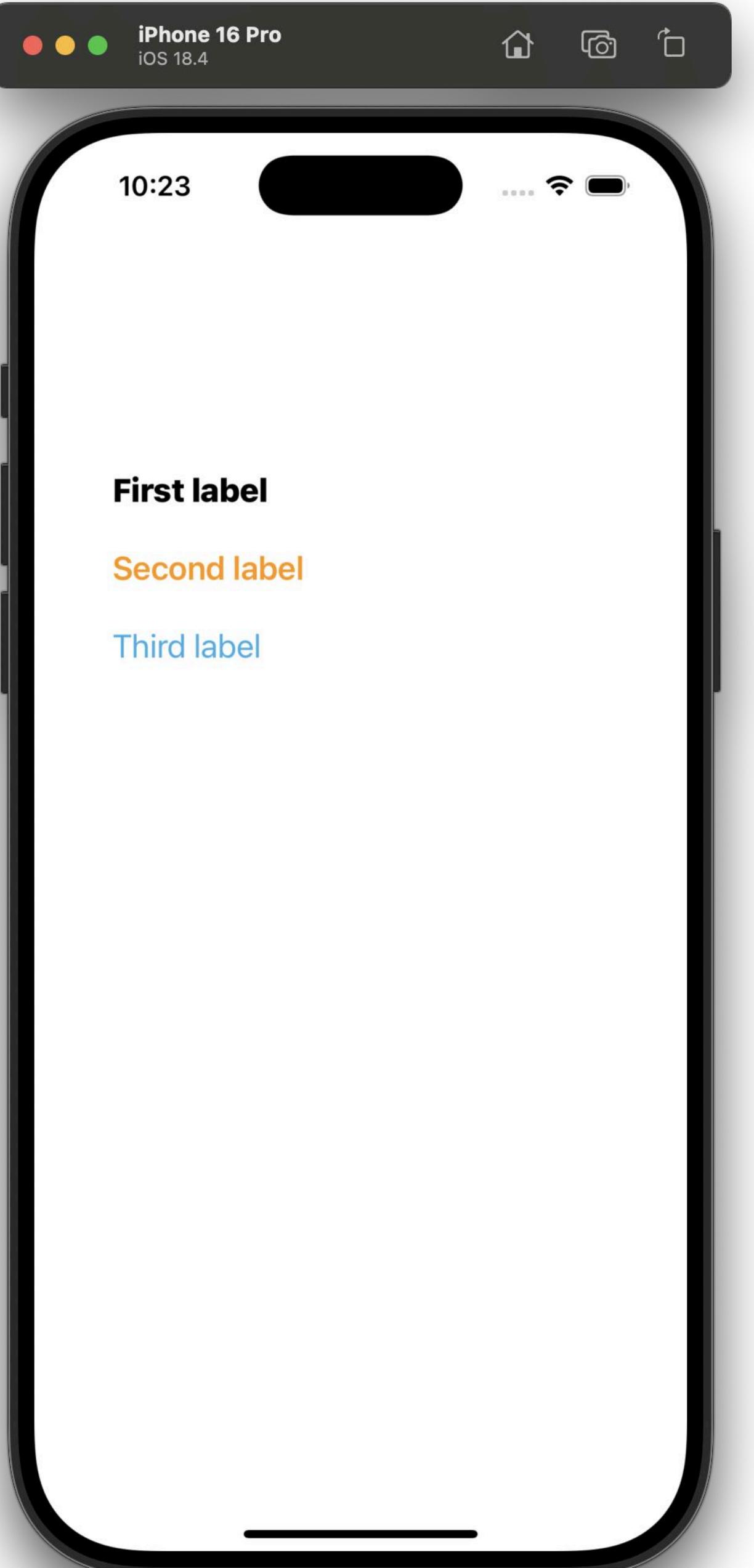
- leadingAnchor & trailingAnchor
 - 오른쪽 → 왼쪽으로 읽는 언어에서는 좌 / 우가 뒤집힐 수 있음
- leftAnchor & rightAnchor
 - 절대적인 좌 / 우
- Apple에서는 leading & trailing의 사용을 권장하는 편

view.safeAreaLayoutGuide

- Navigation Bar, Tab Bar, Toolbar, 그 외의 다른 상위 view에
가려지지 않는 영역



UIStackView

 UIStackView



UIStackView

```
35  override func viewDidLoad() {
36      super.viewDidLoad()
37      view.backgroundColor = .white
38      view.addSubview(firstLabel)
39      view.addSubview(secondLabel)
40      view.addSubview(thirdLabel)
41      firstLabel.translatesAutoresizingMaskIntoConstraints = false
42      secondLabel.translatesAutoresizingMaskIntoConstraints = false
43      thirdLabel.translatesAutoresizingMaskIntoConstraints = false
44      NSLayoutConstraint.activate([
45          firstLabel.topAnchor.constraint(equalTo: view.safeAreaLayoutGuide.topAnchor, constant: 108),
46          firstLabel.leadingAnchor.constraint(equalTo: view.safeAreaLayoutGuide.leadingAnchor, constant: 48),
47          firstLabel.trailingAnchor.constraint(equalTo: view.safeAreaLayoutGuide.trailingAnchor, constant: -48),
48
49          secondLabel.topAnchor.constraint(equalTo: firstLabel.bottomAnchor, constant: 24),
50          secondLabel.leadingAnchor.constraint(equalTo: firstLabel.leadingAnchor),
51          secondLabel.trailingAnchor.constraint(equalTo: firstLabel.trailingAnchor),
52
53          thirdLabel.topAnchor.constraint(equalTo: secondLabel.bottomAnchor, constant: 24),
54          thirdLabel.leadingAnchor.constraint(equalTo: firstLabel.leadingAnchor),
55          thirdLabel.trailingAnchor.constraint(equalTo: firstLabel.trailingAnchor),
56      ])
57 }
```

NoStackViewController.swift

UIStackView

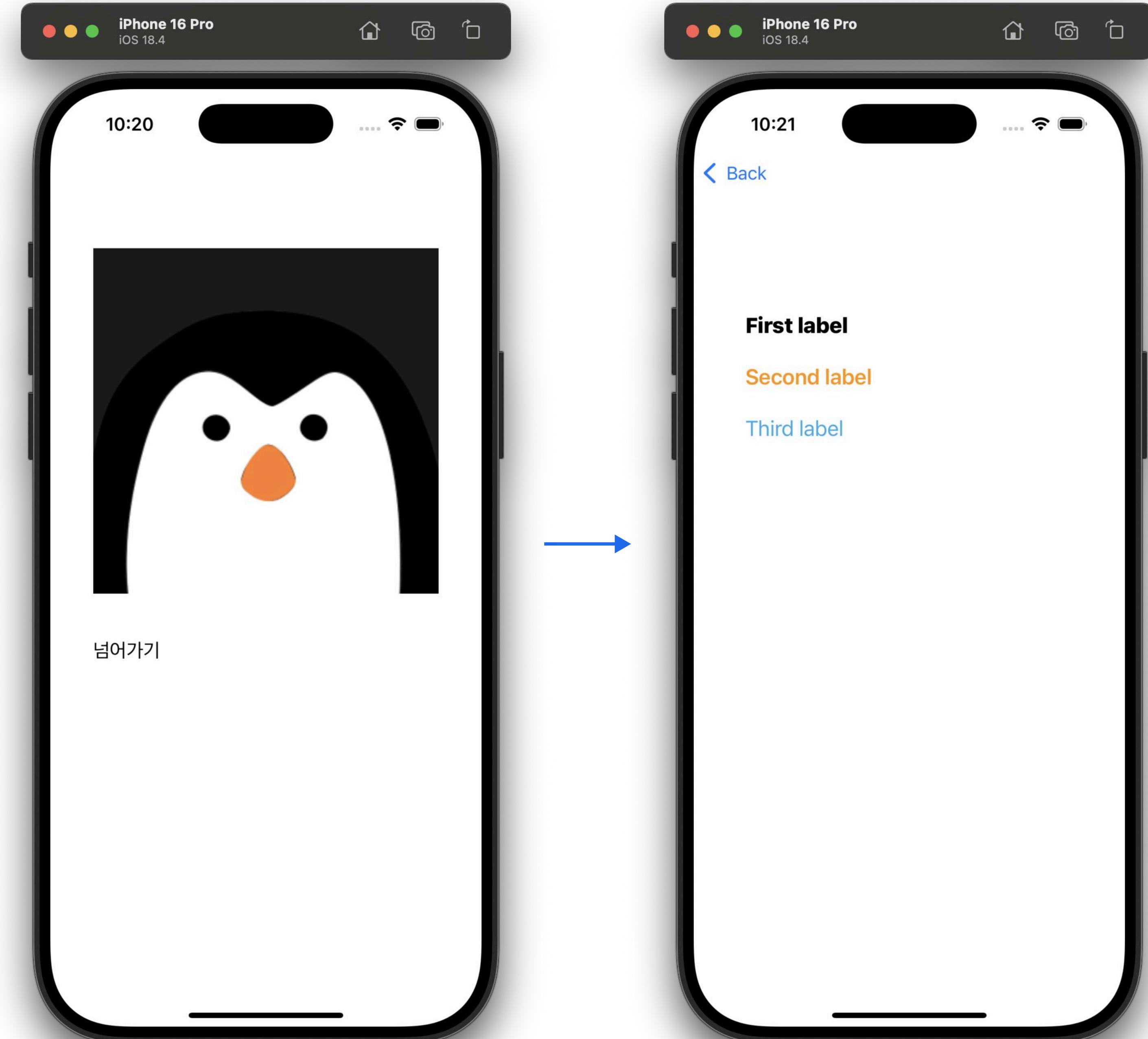
- 일정한 간격으로 View를 나열해야 할 때 유용 ([공식 문서](#))
- `axis` 설정에 따라 가로 혹은 세로로 View 나열 가능

```
24  private lazy var labelStackView: UIStackView = {
25      let stackView = UIStackView(arrangedSubviews: [firstLabel, secondLabel])
26      stackView.spacing = 48
27      stackView.axis = .vertical
28      return stackView
29  }()
30
31  override func viewDidLoad() {
32      super.viewDidLoad()
33      // Do any additional setup after loading the view.
34      view.backgroundColor = .white
35      view.addSubview(labelStackView)
36      labelStackView.translatesAutoresizingMaskIntoConstraints = false
37      NSLayoutConstraint.activate([
38          labelStackView.leadingAnchor.constraint(equalTo: view.safeAreaLayoutGuide.leadingAnchor, constant: 24),
39          labelStackView.trailingAnchor.constraint(equalTo: view.safeAreaLayoutGuide.trailingAnchor, constant: -24),
40          labelStackView.topAnchor.constraint(equalTo: view.safeAreaLayoutGuide.topAnchor, constant: 32)
41      ])
42  }
```

StackViewController.swift

Navigation

Navigation



UIViewController

- UIKit에서의 UIViewController → UIView + Controller
 - 하나의 root view를 관리: root view는 하나 이상의 subview로 구성될 수 있다
 - 하나의 앱은 최소 하나 이상의 UIViewController로 구성된다

UINavigationController

- 두 개의 UIViewController 연결
 - 아까 SceneDelegate에서 작성했던 코드의 의미
 - ```
window?.rootViewController = UINavigationController(rootViewController: ViewController())
window?.makeKeyAndVisible()
```
- 내부적으로 Stack을 이용하여 다른 UIViewController를 push/pop
  - “< 뒤로” 아이콘이 있는 화면들

# **Life Cycle**

```
import UIKit

class ViewController: UIViewController {

 override func viewDidLoad() {
 super.viewDidLoad()
 // Do any additional setup after loading the view.
 }

 override func view
}
```

**M** `viewDidAppear(_ animated:)`

**M** `viewWillAppear(_ animated:)`

**M** `viewIsAppearing(_ animated:)`

**M** `viewDidDisappear(_ animated:)`

**M** `viewWillDisappear(_ animated:)`

**M** `viewDidLayoutSubviews()`

**M** `viewWillLayoutSubviews()`

**M** `viewLayoutMarginsDidChange()`

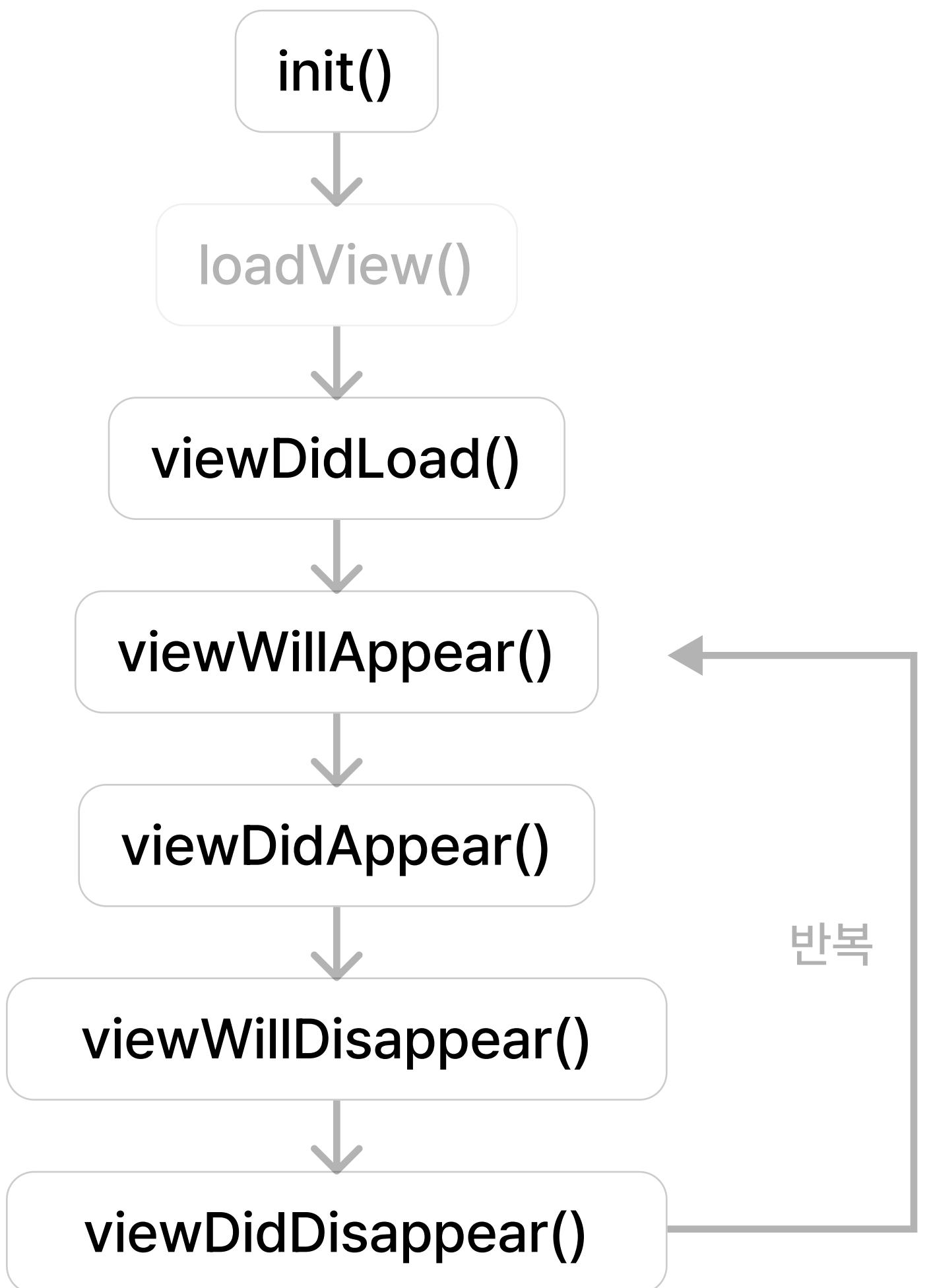
**M** `viewSafeAreaInsetsDidChange()`

**M** `viewDidAppear(_ animated: Bool)`

Notifies the view controller that its view was added to a view hierarchy.

# UIViewController의 Life Cycle

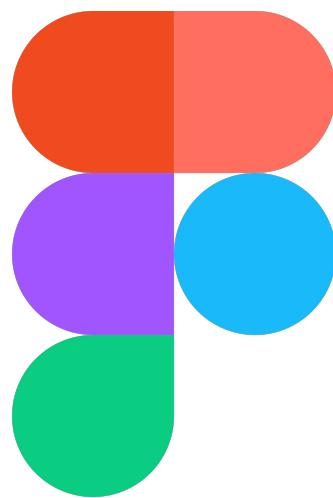
- 우리는 viewDidLoad() ~ viewDidDisappear()에 집중
  - viewDidLoad() : view hierarchy가 메모리에 load되었을 때
    - 편의상 딱 한 번만 실행되는 구간이라고 생각해도 된다
    - 어떤 작업을 해야할까?
  - 각 메서드는 자동으로 호출된다
    - 개발자가 할 일: 적당한 메서드를 override하여 원하는 타이밍에 작업 수행



# Figma

# Figma란?

- 대표적인 UI/UX 협업툴 중 하나
  - 기획자, 디자이너, 개발자
- 브라우저 및 데스크톱 앱 모두 지원
  - 설치 링크(선택): <https://www.figma.com/ko-kr/downloads/>



# 개발자 세미나에서 왜 Figma를 다루나요?

- 개발자는 혼자 일하지 않습니다
  - 디자이너가 제공하는 스펙에 맞는 UI를 구현
  - QA와 유지보수에 들어가는 비용 줄이기
    - ex. 똑같은 Figma 파일, 서로 다른 안드로이드/iOS 화면

수강신청 빈자리 알림 서비스 창을 들어갔을 때, 수강신청 사이트 버튼이 맨 아래 강좌(6과목 담은 기준) 수강인원 수를 가려서 조금 불편한 것 같아요!

- 세미나의 모든 과제는 Figma 파일을 제공합니다
  - 직접 Layout을 확인하고, 그대로 구현해주세요

| 빈자리 알림                        |         | 편집 |
|-------------------------------|---------|----|
| ● 지구환경과학부, 3학년                | 22명/40명 |    |
| ● 금 11:00~11:50/금 13:00~16:50 |         |    |
| ● 500-L311                    |         |    |
| <b>지질도학 실험</b>                | 우주선/3학점 |    |
| ● 지구환경과학부, 3학년                | 15명/40명 |    |
| ● 월 09:00~11:50/수 09:00~11:50 |         |    |
| ● 25-1-406                    |         |    |
| <b>대기화학개론 및 실습</b>            | 박록진/3학점 |    |
| ● 지구환경과학부, 4학년                | 15명/40명 |    |
| ● 월 10:00~11:50/수 10:00~11:50 |         |    |
| ● 500-L310                    |         |    |
| <b>대기수치모델링 개론 및 실습</b>        | 박성수/3학점 |    |
| ● 지구환경과학부, 4학년                | 6명/40명  |    |
| ● 화 12:30~14:20/목 12:30~14:20 |         |    |
| ● 500-L311 / 25-1-418         |         |    |
| <b>대기역학 2</b>                 | 손석우/3학점 |    |
| ● 지구환경과학부, 3학년                | 18명/40명 |    |
| ● 화 09:30~10:45/목 09:30~10:45 |         |    |
| ● 500-L311                    |         |    |
| <b>대기물리 2</b>                 | 백종진/3학점 |    |
| ● 지구환경과학부, 3학년                | 25명/40명 |    |
| ● 월 12:30~13:45/수 12:30~13:45 |         |    |
| ● 500-L304                    |         |    |
| <b>위성기상기후학</b>                | 이상무/3학점 |    |
| ● 지구환경과학부, 4학년                |         |    |
| ● 화 15:00~16:50/목 15:00~16:50 |         |    |
| ● 25-1-418                    |         |    |

수강신청 사이트

# 개발자 세미나에서 왜 Figma를 다루나요?

- 개발자는 혼자 일하지 않습니다
  - 디자이너가 제공하는 스펙에 맞는 UI를 구현
  - QA와 유지보수에 들어가는 비용 줄이기
    - ex. 똑같은 Figma 파일, 서로 다른 안드로이드/iOS 화면

수강신청 빈자리 알림 서비스 창을 들어갔을 때, 수강신청 사이트 버튼이 맨 아래 강좌(6과목 담은 기준) 수강인원 수를 가려서 조금 불편한 것 같아요!

- 세미나의 모든 과제는 Figma 파일을 제공합니다
  - 직접 Layout을 확인하고, 그대로 구현해주세요



# Figma 보는 방법 - Font, Color

**Black(900)**

**ExtraBold(800)**

**Bold(700)**

**SemiBold(600)**

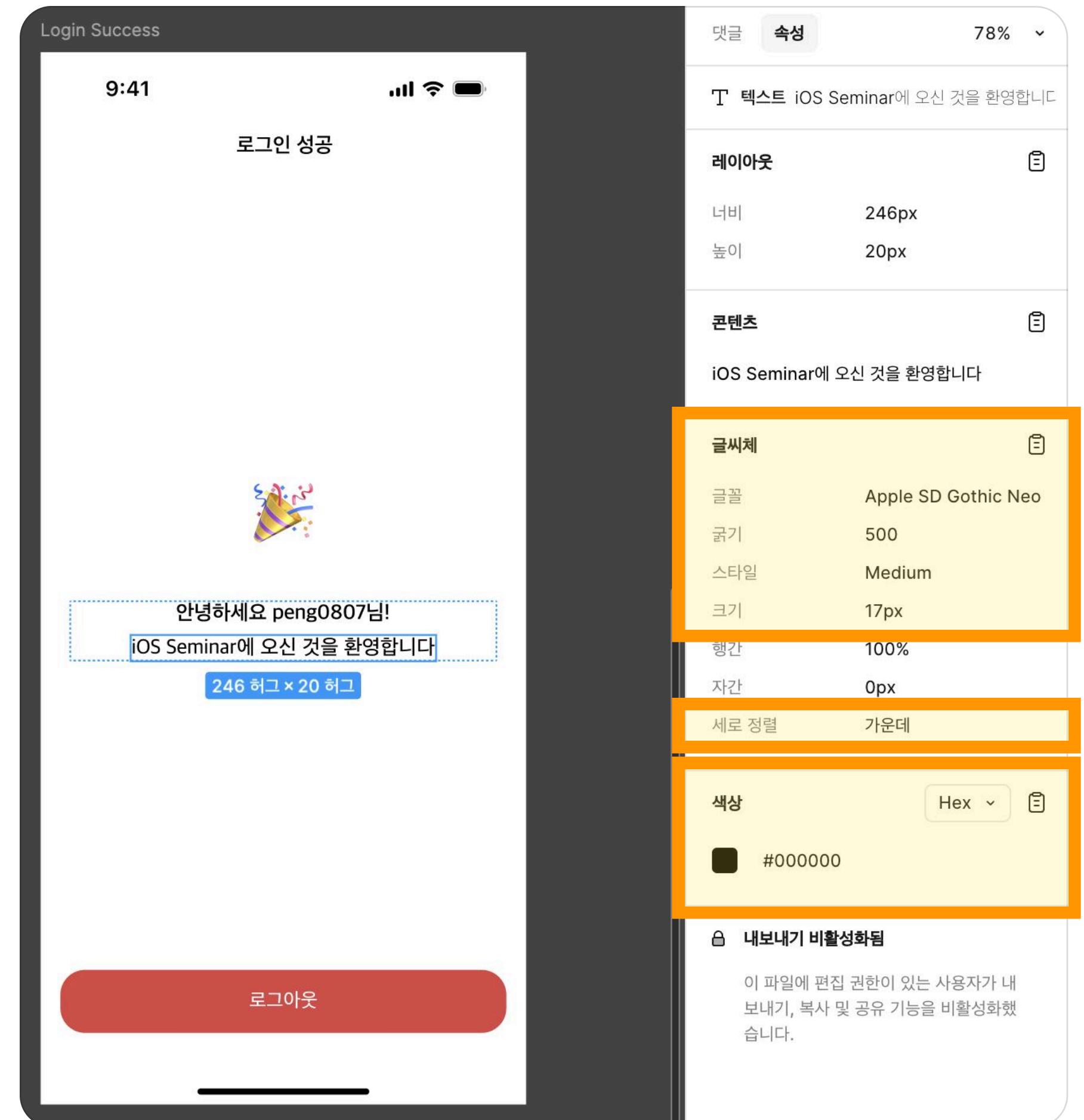
**Medium(500)**

\*Regular(400) : 기본값

Light(300)

ExtraLight(200)

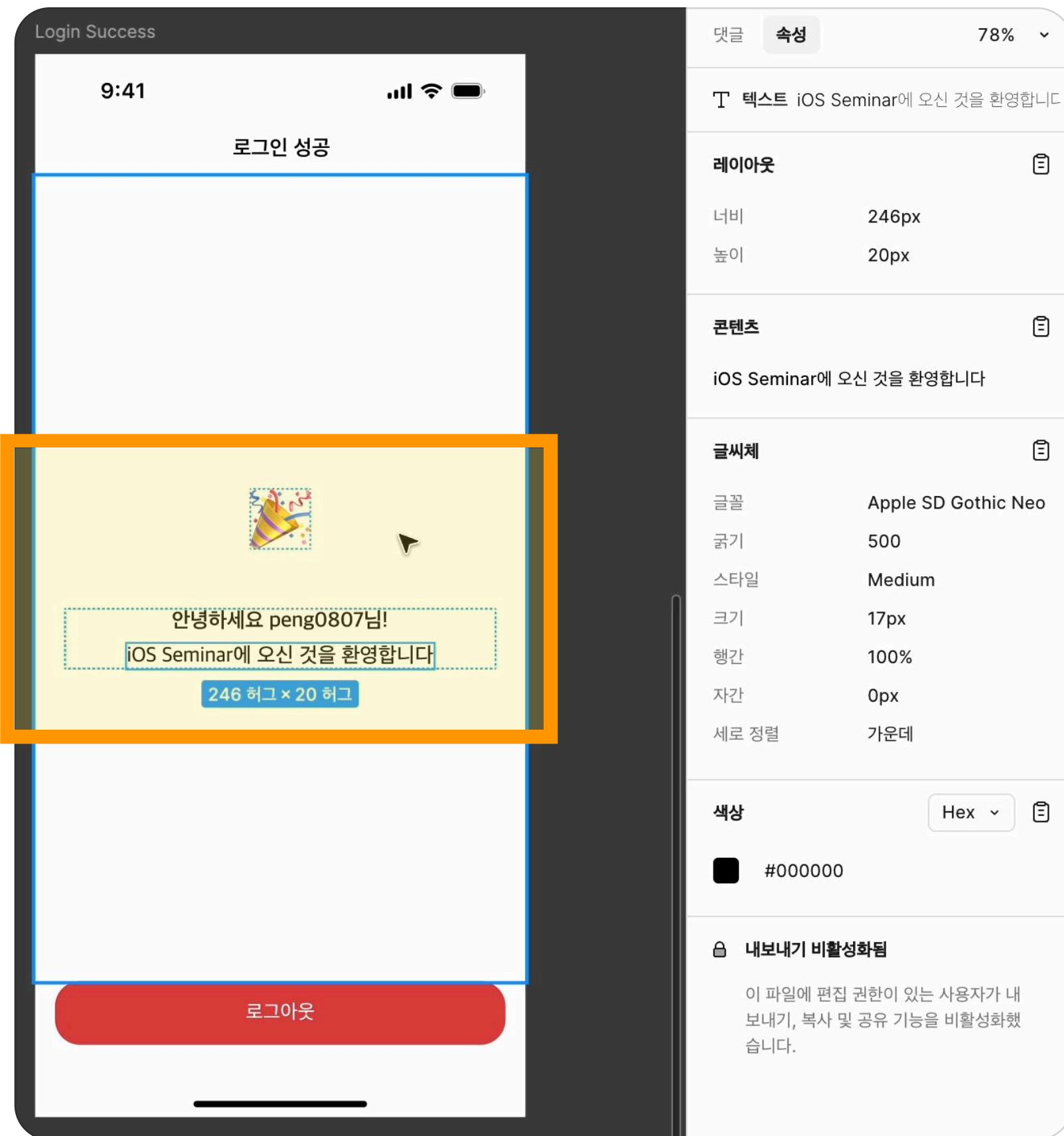
Thin(100)



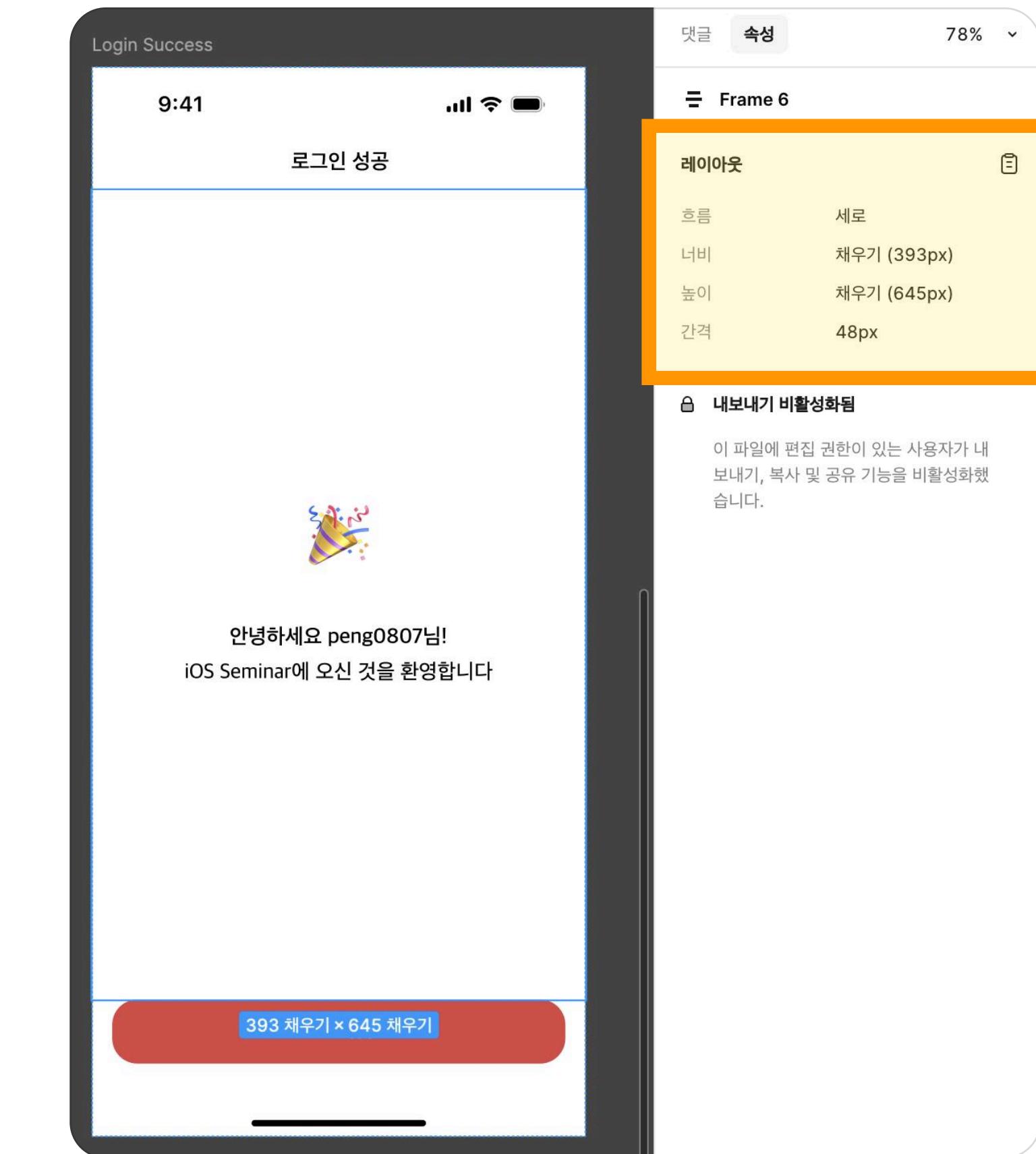
Figma - Font, Color

# Figma 보는 방법 - Constraint, Layout

- Constraint: option 키 + 마우스 hover



Figma - Constraint

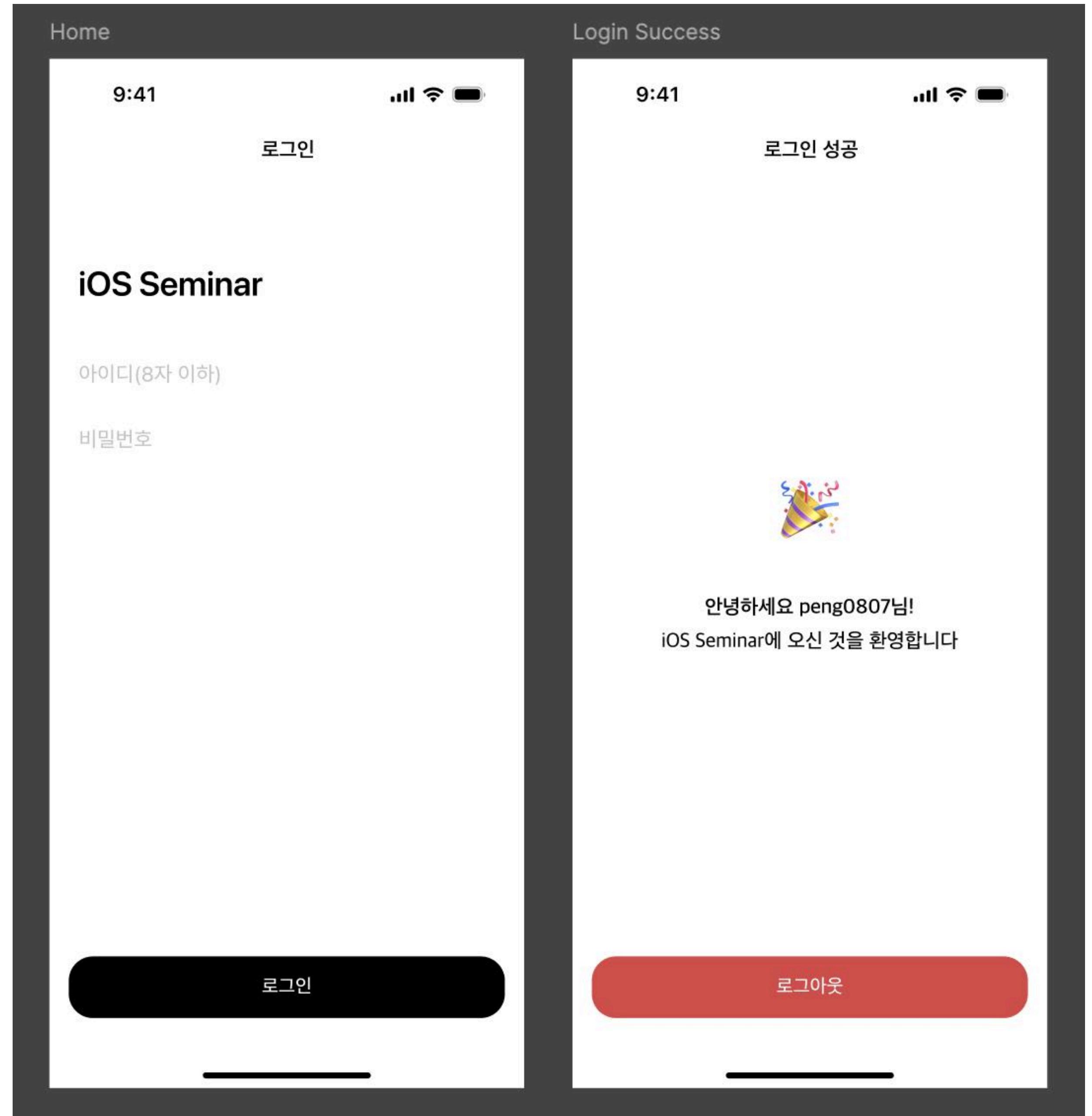


Figma - Layout

# 과제 0 안내

## 간단한 로그인 View 구현

- 과제 0 디자인 링크(Figma)
- 과제 0 스켈레톤 코드 링크(Github)
- 과제 목표
  - No Storyboard
    - Auto Layout
    - UITextField
    - UIButton
    - UIStackView
    - UINavigationController



## 과제 상세 스펙 및 주의사항 (1)

- 전체
  - Storyboard를 사용하지 않습니다 (LaunchScreen.storyboard 제외)
  - 모든 Text, Color, Constraint, Corner Radius 등은 피그마에 주어진 값을 사용합니다
  - Figma에 스펙이 누락된 부분이 있다면, 슬랙에서 질문 부탁드립니다
- 스켈레톤 코드
  - 스켈레톤 코드만으로는 과제를 해결하실 수 없습니다. 적절한 변수와 함수를 추가해 주세요
  - 스켈레톤 코드는 자유롭게 수정 가능하며, 아예 사용하지 않으셔도 괜찮습니다
    - 사용 여부는 과제 채점에 아무런 영향을 미치지 않습니다

## 과제 상세 스펙 및 주의사항 (2)

- Font
  - Figma에 표시되는 폰트명 SF Pro / Apple SD Gothic Neo는 Apple의 시스템 폰트이므로 폰트의 size, weight, color만 고려합니다
- UITextField
  - 모든 UITextField의 height는 기본값을 사용합니다 (따로 설정 필요X)
  - 모든 UITextField는 입력하는 동안 오른쪽에 x 버튼이 뜨도록 합니다
  - 비밀번호를 입력하는 UITextField는 입력값이 ●로 마스킹되도록 설정합니다
- UIButton
  - 로그인: id가 조건을 만족하고 password가 비어 있지 않은 경우에만 다음 화면으로 넘어갑니다
  - 로그아웃: 버튼을 눌러 첫 화면으로 되돌아올 때, 모든 상태가 초기화되도록 합니다
    - (아이디 및 비밀번호 UITextField 입력값 등)

## 과제 Bonus 스펙

- 아이디를 입력하는 UITextField에 영어를 입력하면 자동으로 첫 글자가 대문자가 됩니다. 어떻게 해결할 수 있을까요?
- 마지막으로 로그인할 때 사용했던 아이디는 다음으로 앱을 켤 때 자동으로 아이디 UITextField에 채워져 있도록 해봅시다.
  - hint: UserDefaults

## 과제 제출을 위한 세팅

1. 슬랙 "#ios-공지"에 올라온 <https://classroom.github.com/> 링크를 클릭
  - a. 초대를 받으면 자동으로 `ios-assignment-{github name}`라는 이름의 private repository가 만들어짐
2. 과제를 진행할 로컬 위치에 repository를 적절하게 clone
3. 매 과제를 진행할 때는 `main` 브랜치로부터 `assignment0`, `assignment1`, ... 브랜치를 각각 생성하여 작업
4. 완료한 과제는 `main` 브랜치로 `Pull request` 추가하여 제출
  - a. PR 생성 시 반드시 세미나장을 Reviewers로 지정
  - b. 모든 회차의 세미나가 종료되기 전까지는 'Merge pull request' 누르시면 안됩니다
5. 커밋(commit)은 자잘하게 쪼갤 것을 권장
  - a. "finished assignment 0" << 이렇게 하나로 통치지 말아주세요

```
-o Main.storyboard 삭제
-o SceneDelegate 변경 (검은 화면)
-o SecondViewController 생성
-o loginButton 생성
-o loginButton 디자인 반영
```