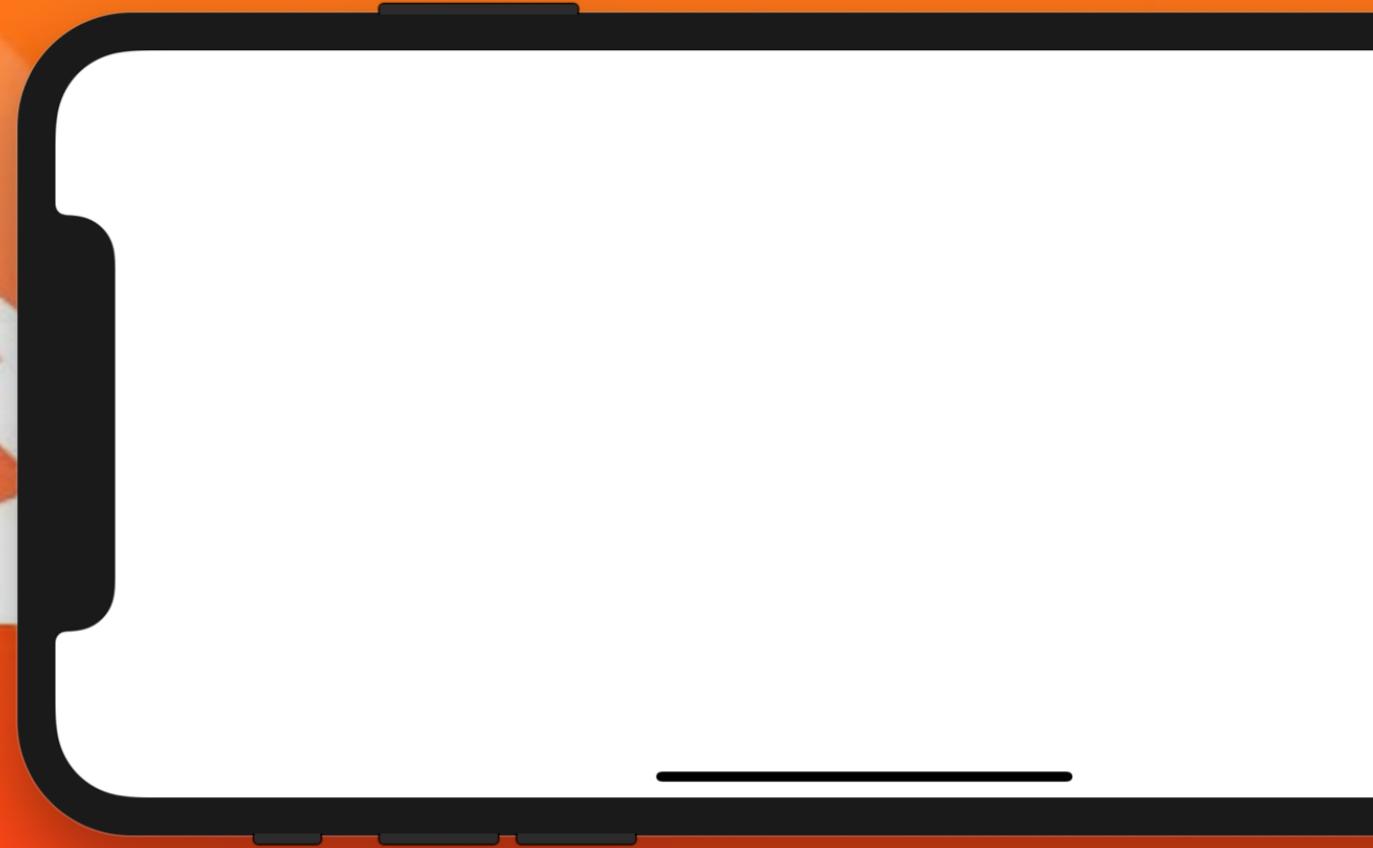


스위프트를  
이용한  
아이폰  
앱 개발  
시작하기



미니 프로젝트 | DAY #01

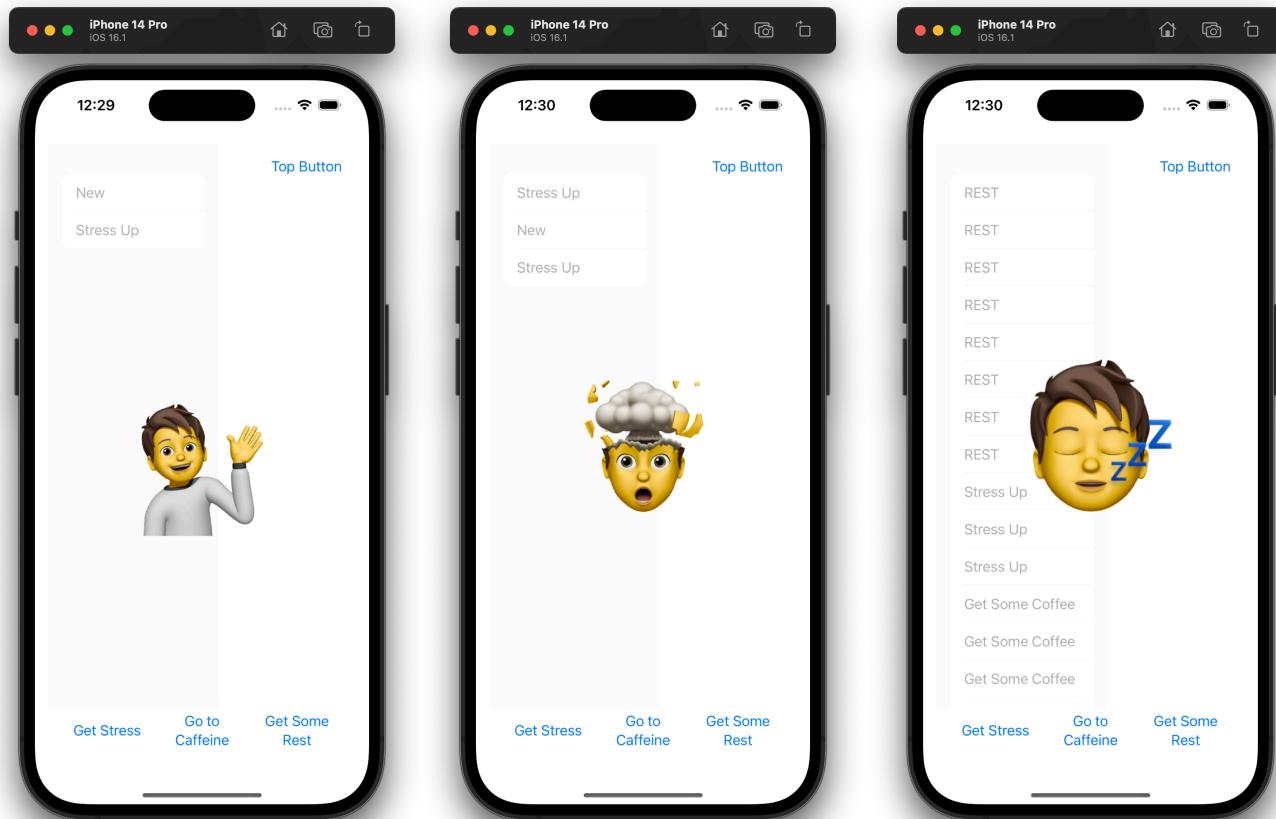
# Overview

# Overview

# 미니 프로젝트 개요

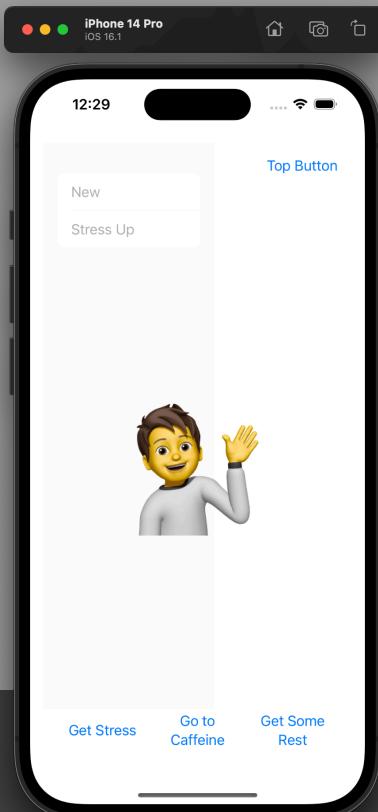
Mini-Term  
Project #1  
Overview

- 커피/스트레스/휴식에 따라서 현재의 상태를 재미있게 나타내는 방법을 아이디어로 구현한 싱글뷰 앱



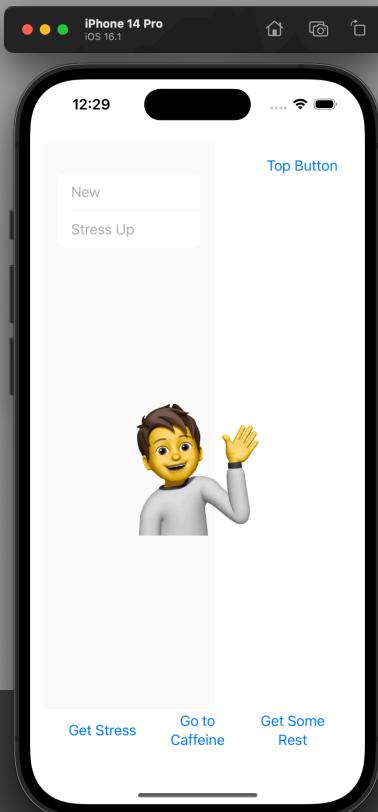
미니 프로젝트 | DAY #01

## 주요 요구사항



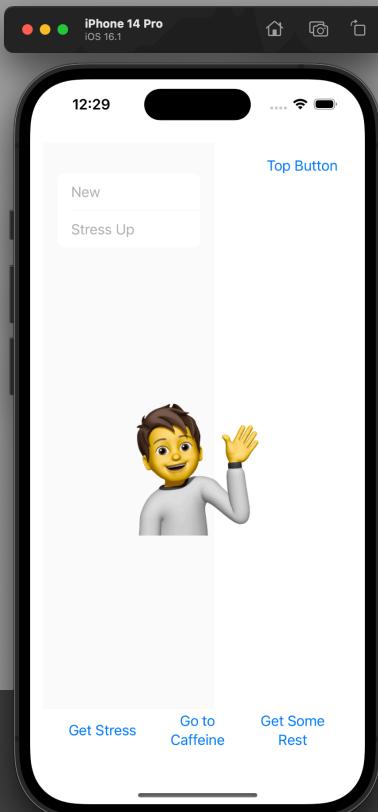
- 화면 가운데에는 사람 모양의 이미지를 배치한다
- (현실의) 사용자가 내 상태에 따라 버튼 누름
  - 지금 스트레스를 받으면 스트레스 버튼 클릭
  - 커피 마시면 커피 버튼 클릭
  - 휴식을 취하면 휴식 버튼 클릭
  - 초기화 버튼을 누르면, 처음 상태로 돌아가기
- 상태에 따라서 사람의 이미지가 변경될 수 있음
- 변경한 상태의 목록을 저장하고 출력해야 함

## 심화 요구사항



- 화면 가운데에는 사람 모양의 이미지를 배치한다
  - 상태가 누적될 경우, 크기가 점점 커지도록 함
- 상태에 따라서 사람의 이미지가 변경될 수 있음
- 상태가 변경될 때 이미지를 자신만의 것으로 변경
- 상태가 변경될 때, 배경 색을 점점 변경
  - (기본) 백색 / 녹색 / 파랑 / 갈색 / 회색 / 검정색
- 변경한 상태의 목록을 저장하고 출력해야 함
  - (추후에) 변경한 상태의 목록을 일자별로 관리

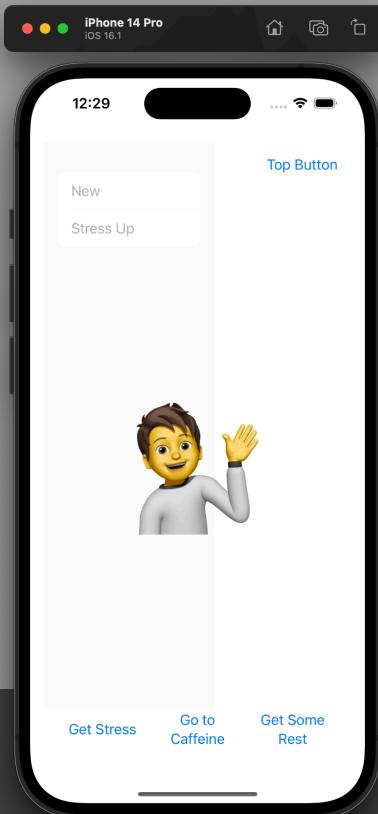
# 학습요소 SwiftUI



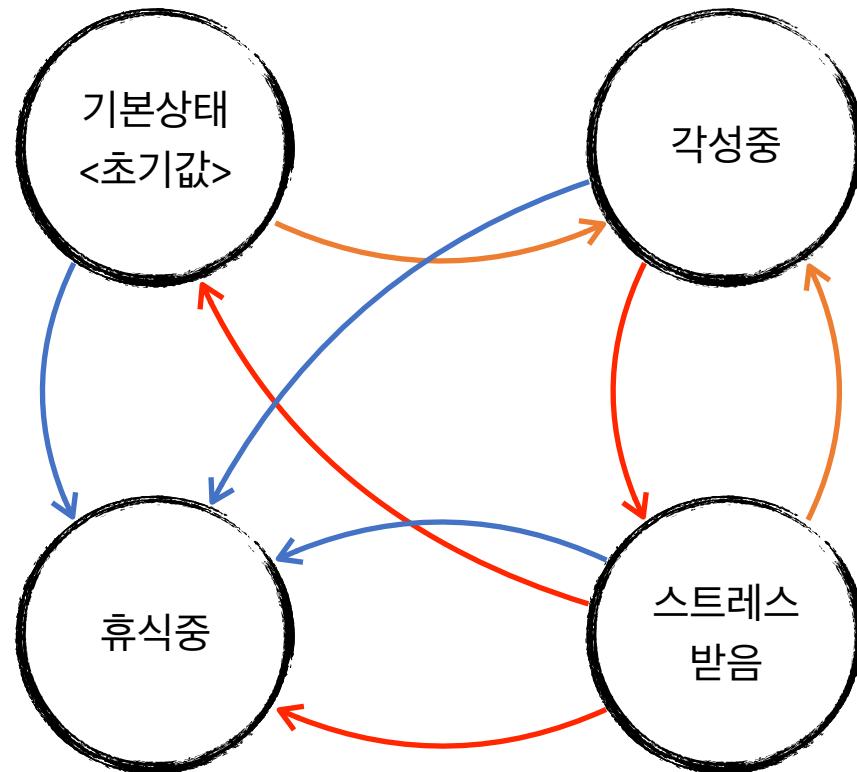
- Stack을 모두 다뤄볼 수 있음
  - H/V/Z Stack 모두 포함
- struct를 필요에 따라서 구분하여 가독성 향상
- Button/ Image/ Text/ List/ Spacer 소개
- @State 를 통해서 화면이 그려질 수 있는 시점 파악
- 그외
  - Array 자료구조 학습
  - 다중 반복 함수 학습
  - enum, Case

미니 프로젝트 | DAY #01

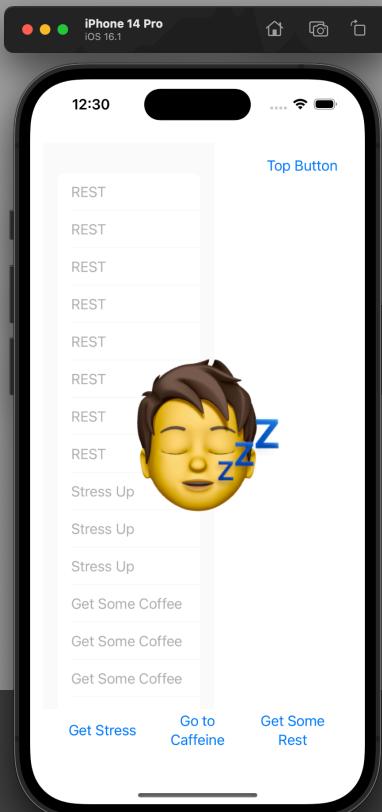
## 모델 소개



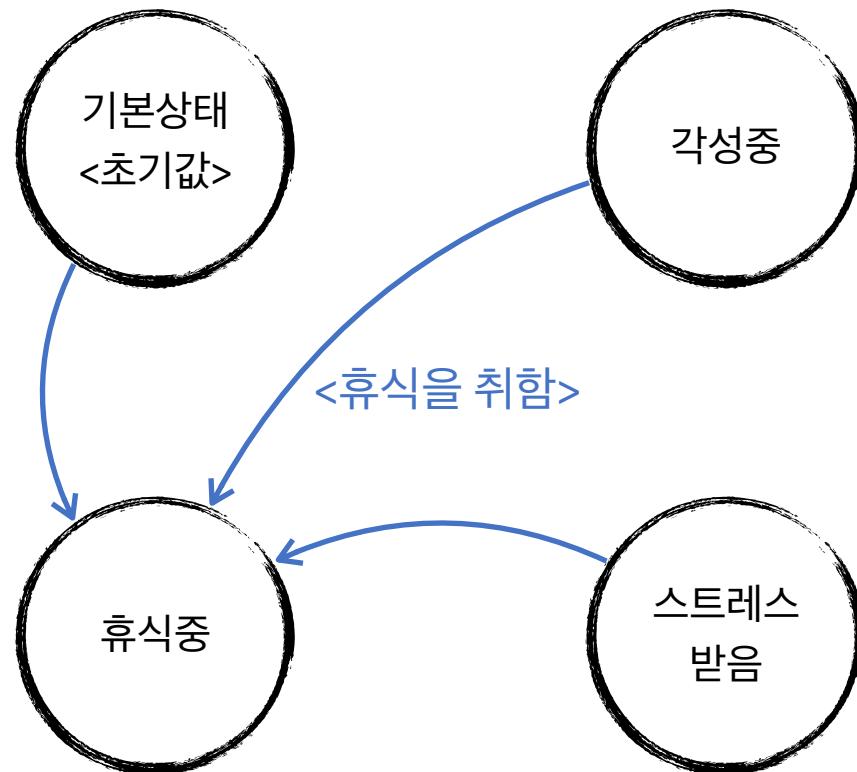
- “카페인홀릭”의 상태 전개도



## 모델 소개



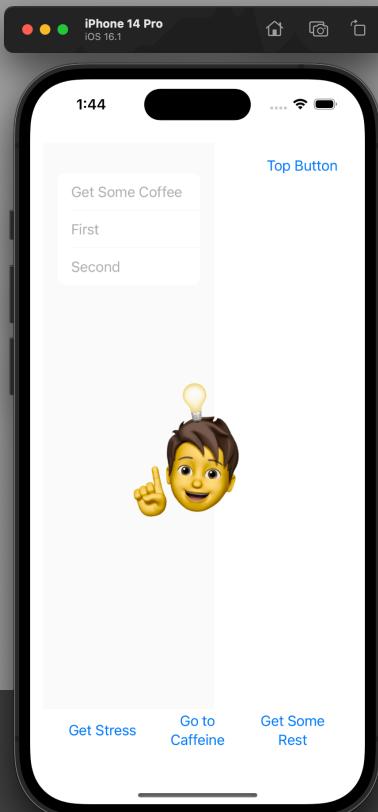
- “카페인홀릭”의 상태 전개도 <1>



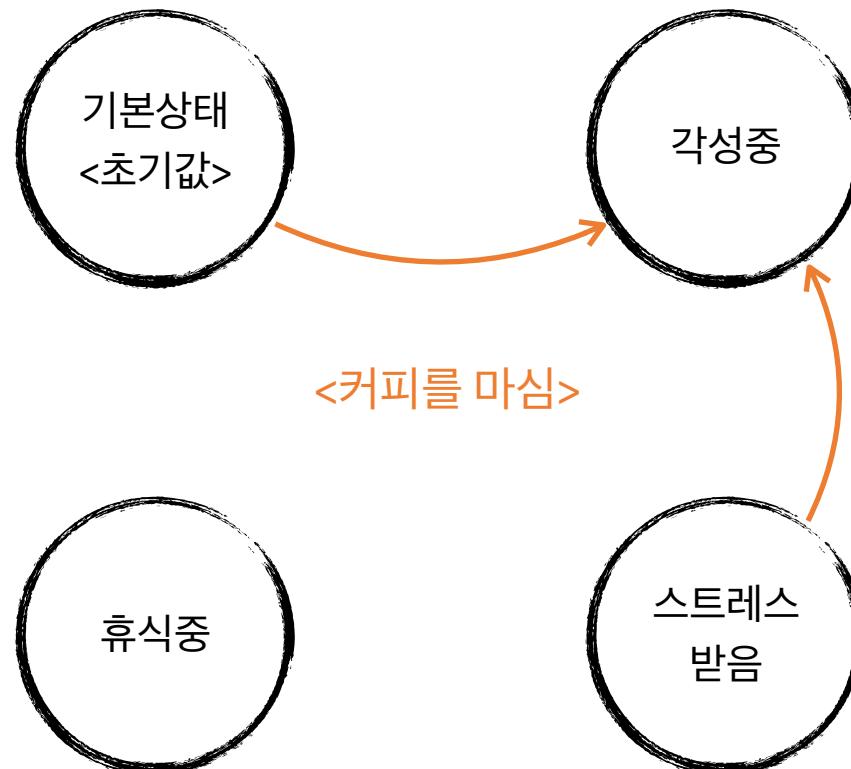
미니 프로젝트

DAY #01

## 모델 소개

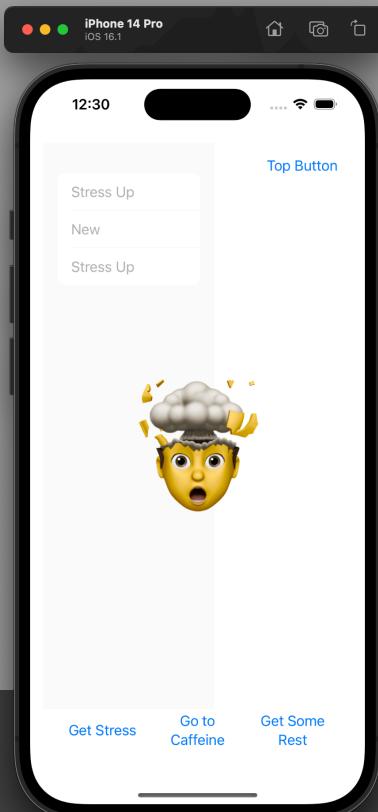


- “카페인홀릭”의 상태 전개도 <2>

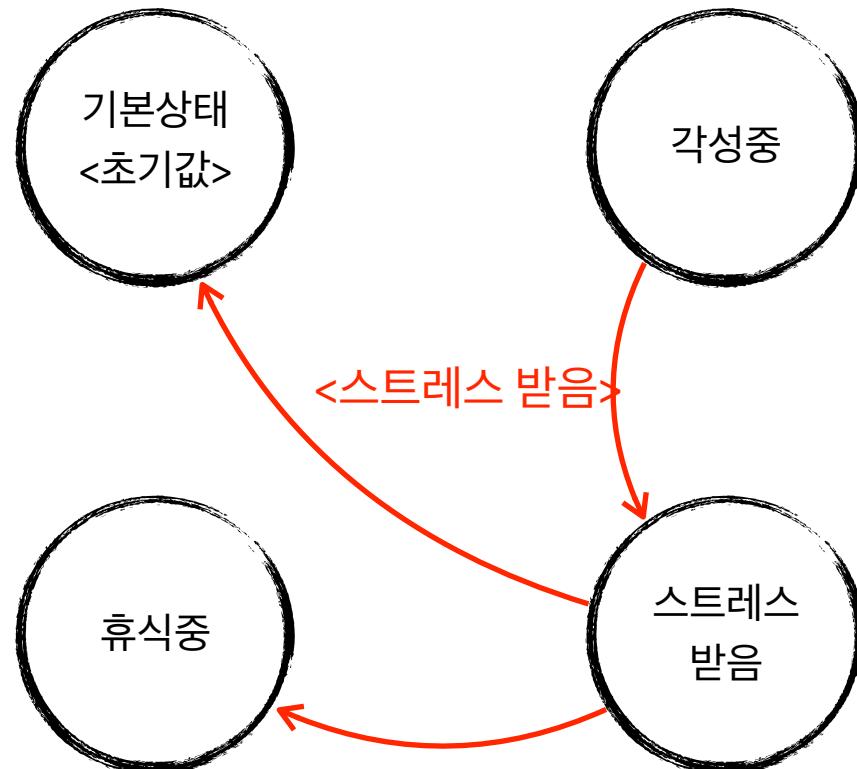


미니 프로젝트 | DAY #01

## 모델 소개



- “카페인홀릭”의 상태 전개도 <3>

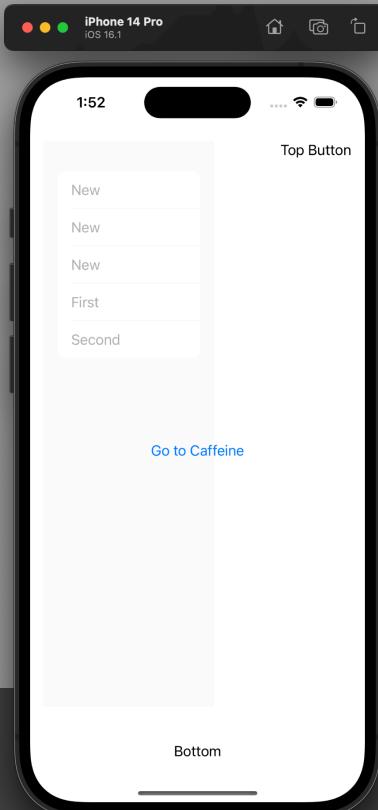


# 각 단계별 목표

#1 ~ #4 단계별 목표 요약

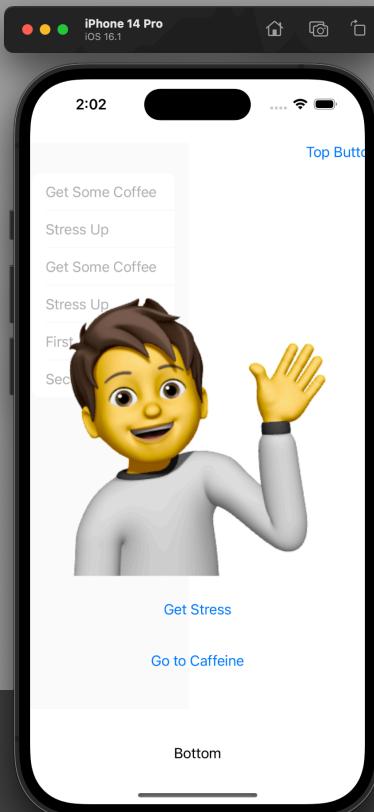
## Step#1 목표

- (기본 화면) Stack을 이용해 기본 화면을 구성한다
  - ZStack을 이용해 계층 생성(일간기록/버튼화면)
  - HStack과 VStack을 이용해 화면 틀을 구성한다
- (버튼 화면) 화면 중앙에 버튼을 만든다
- (일간 기록) 배경에 일일 기록을 위에서부터 아래로 구성하는 목록을 구성한다
- (SwiftUI) @State를 활용해서 목록 화면 새로고침 응답



# 각 단계별 목표

## Step#2 목표

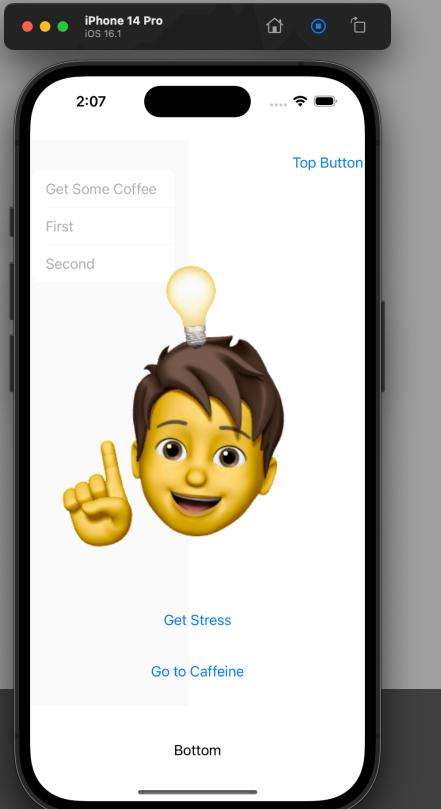


- (버튼 동작) 화면 중앙 버튼을 누르면 배경에 목록에 “Get Some Coffee”를 출력한다
- (화면 구성) “Go to Caffeine” 버튼을 추가한 뒤 배경 목록에 추가하도록 구현한다
- (화면 구성) Image를 추가하고, 원하는 이미지를 설정한다

미니 프로젝트 | DAY #01

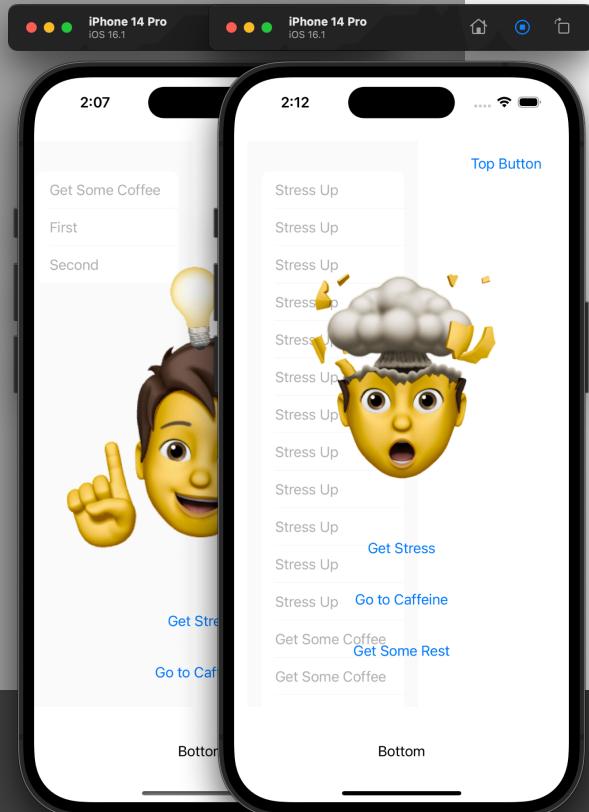
## Step #3 목표

- (코드 분리) 카페인 상태 모델 작성을 위해 “CaffeineModel.swift” 파일 생성 및 작성
- (화면 구성) 기존 화면과 연결된 코드를 상태 모델에 연결
- (화면 구성) 각 상태에 이미지 연결 코드 추가



## Step #4 목표

- 모든 상태로 이동하는 코드 작성
- 현재 상태와 같은 상태를 누를 경우, 점점 크기를 증가함
- 초기 상태로 돌아갈 수 있도록 함



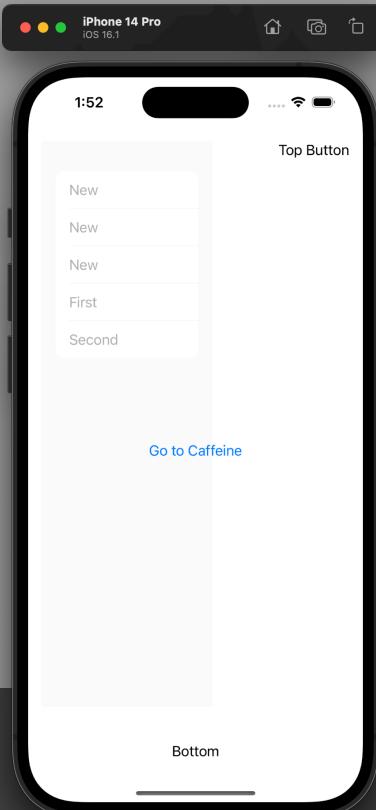
# WalkThrough

코드보며 함께 만들기

# Step #1

기본 UI 구성하기

## Step#1 목표

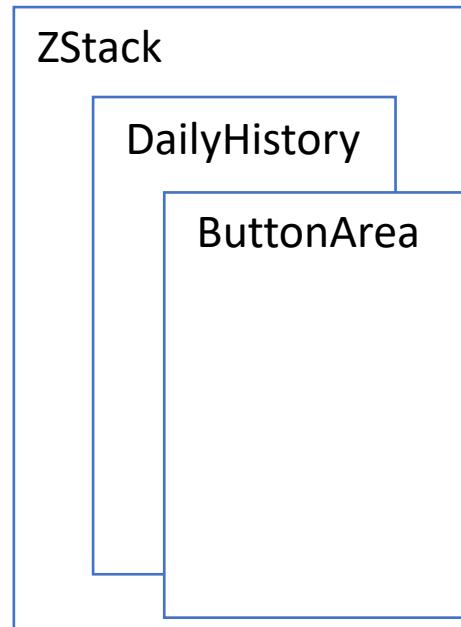
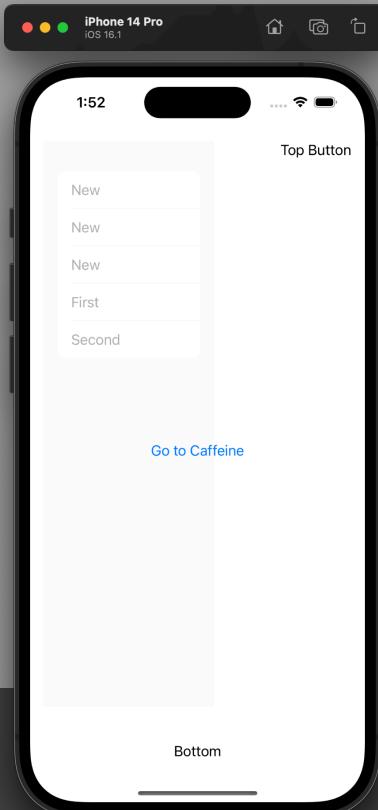


- (기본 화면) Stack을 이용해 기본 화면을 구성한다
  - ZStack을 이용해 계층 생성(일간기록/버튼화면)
  - HStack과 VStack을 이용해 화면 틀을 구성한다
- (버튼 화면) 화면 중앙에 버튼을 만든다
- (일간 기록) 배경에 일일 기록을 위에서부터 아래로 구성하는 목록을 구성한다
- (SwiftUI) @State를 활용해서 목록 화면 새고로침 응답

# Step #1 기본 UI 구성하기

## 기본화면 구성

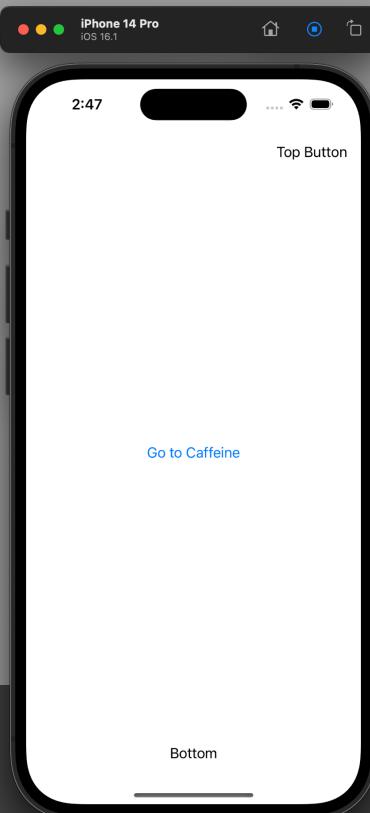
- (기본 화면)Stack을 이용해 기본 화면을 구성한다
  - ZStack을 이용해 계층 생성(일간기록/버튼화면)
  - HStack과 VStack을 이용해 화면 틀을 구성한다



미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

## 기본화면 구성

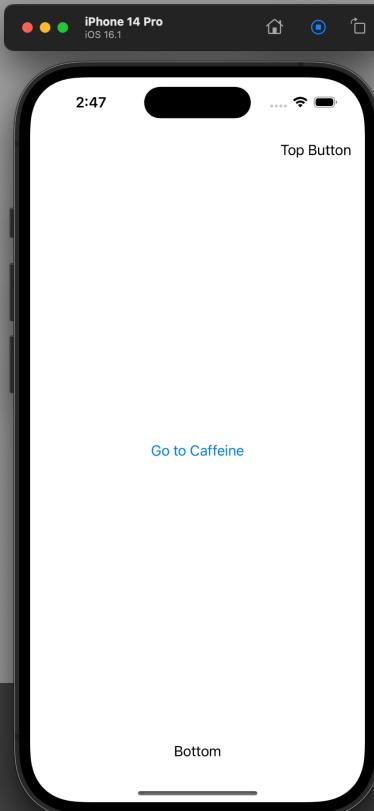


```
var body: some View {
    VStack {
        // Top buttons
        HStack(spacing: 20){
            Text("")
            Spacer()
            Text("Top Button")
        }
        Spacer()
        Button("Go to Caffeine") {
            print("Click")
        }
        Spacer()
        Text("Bottom")
    }
    .padding()
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

## 버튼화면 구성



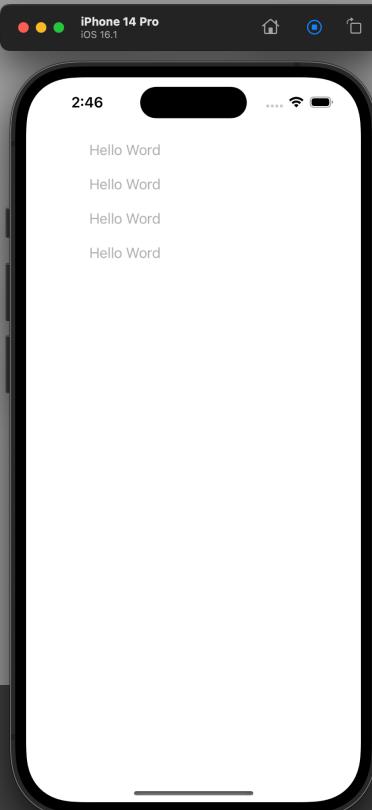
```
var Buttons: some View {  
    VStack {  
        // Top buttons  
        HStack(spacing: 20){  
            Text("")  
            Spacer()  
            Text("Top Button")  
        }  
        Spacer()  
        Button("Go to Caffeine") {  
            print("Click")  
            incList()  
        }  
        Spacer()  
        Text("Bottom")  
    }  
    .padding()  
}
```

```
var body: some View {  
    Buttons  
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

## 일간기록 구성

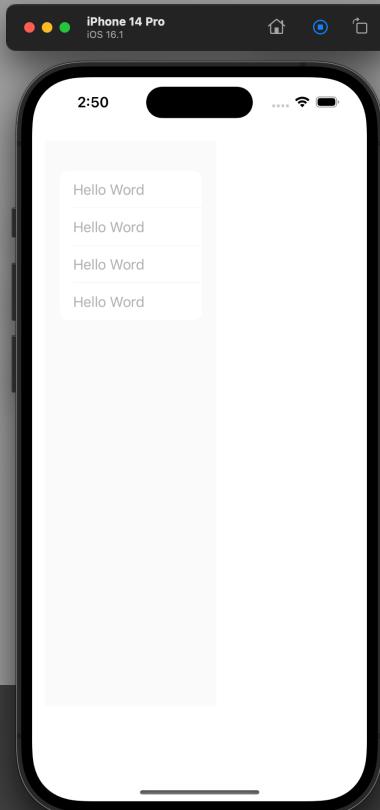


```
var body: some View {
    HStack(alignment: .top) {
        VStack(alignment: .leading, spacing: 20) {
            Text("Hello Word")
            Text("Hello Word")
            Text("Hello Word")
            Text("Hello Word")
            Spacer()
            Text("") // End of Daily Item List
        }
        .frame(width: 200)
        .opacity(0.3)
        Spacer()
    }
    .padding()
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

## 일간기록 구성

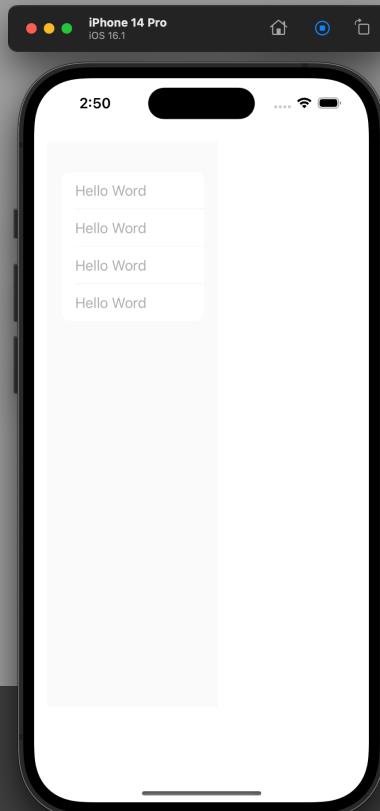


```
var body: some View {
    HStack(alignment: .top) {
        VStack(alignment: .leading, spacing: 20) {
            List {
                Text("Hello Word")
                Text("Hello Word")
                Text("Hello Word")
                Text("Hello Word")
            }
            Spacer()
            Text("") // End of Daily Item List
        }
        .frame(width: 200)
        .opacity(0.3)
        Spacer()
    }
    .padding()
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

## 일간기록 구성

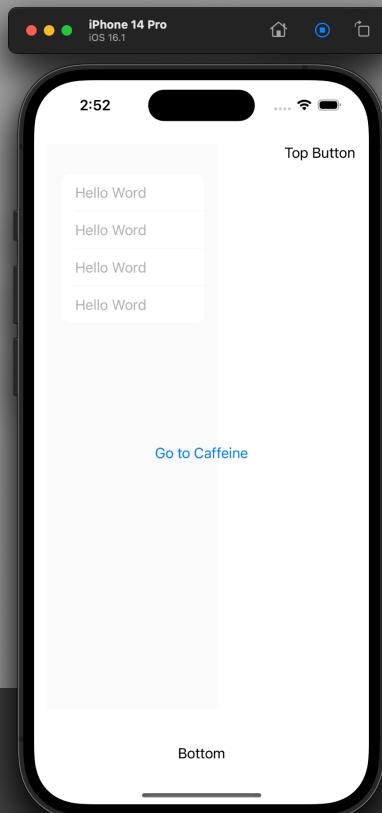


```
var DailyView: some View {
    // Daily History area
    HStack(alignment: .top) {
        VStack(alignment: .leading, spacing: 20) {
            List {
                Text("Hello Word")
                Text("Hello Word")
                Text("Hello Word")
                Text("Hello Word")
            }
            Spacer()
            Text("") // End of Daily Item List
        }
        .frame(width: 200)
        .opacity(0.3)
        Spacer()
    }
    .padding()
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

## 기본화면 통합

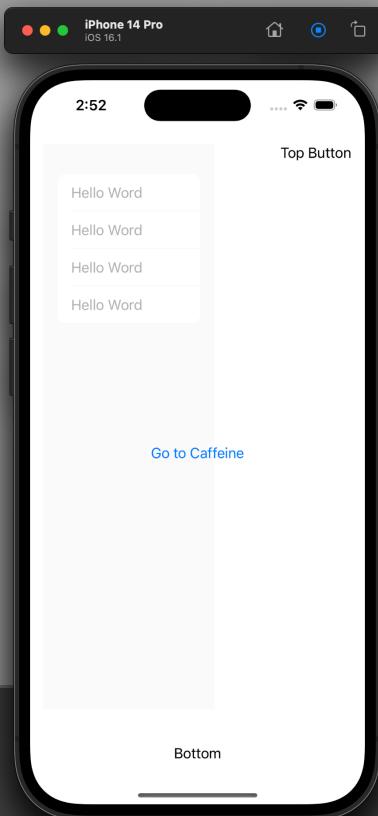


```
var body: some View {  
    ZStack {  
        DailyView  
        Buttons  
    }  
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

화면  
새로고침

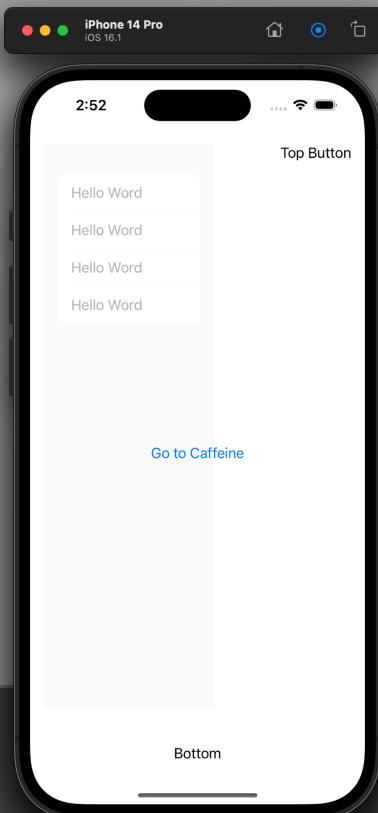


```
struct ContentView: View {  
    @State private var dailyList = ["First", "Second"]  
  
    func incList() {  
        dailyList.insert("New", at: 0)  
    }  
  
    var DailyView: some View {  
        // Daily History area  
        HStack(alignment: .top) {  
            VStack(alignment: .leading, spacing: 20) {  
                List(dailyList, id: \.self) {item in  
                    Text(item)  
                }  
                Spacer()  
                Text("") // End of Daily Item List  
            }  
            .frame(width: 200)  
            .opacity(0.3)  
            Spacer()  
        }  
        .padding()  
    }  
}
```

미니 프로젝트 | DAY #01

# Step #1 기본 UI 구성하기

화면  
새로고침



```
var Buttons: some View {
    VStack {
        // Top buttons
        HStack(spacing: 20){
            Text("")
            Spacer()
            Text("Top Button")
        }
        Spacer()
        Button("Go to Caffeine") {
            print("Click")
            incList()
        }
        Spacer()
        Text("Bottom")
    }
    .padding()
}
```

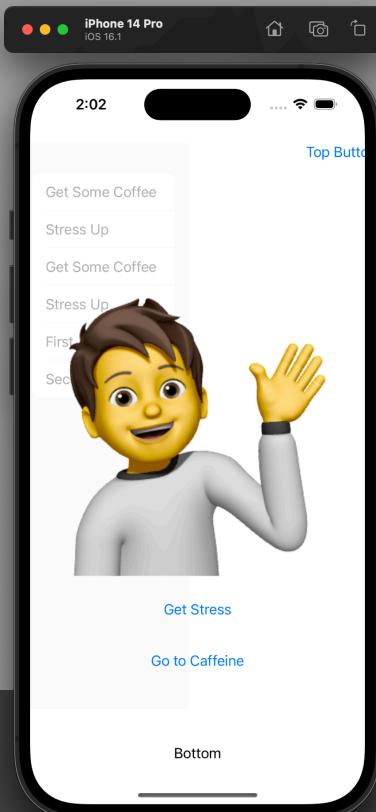
미니 프로젝트 | DAY #01

# Step #2

버튼 동작 만들기

# 각 단계별 목표

## Step#2 목표



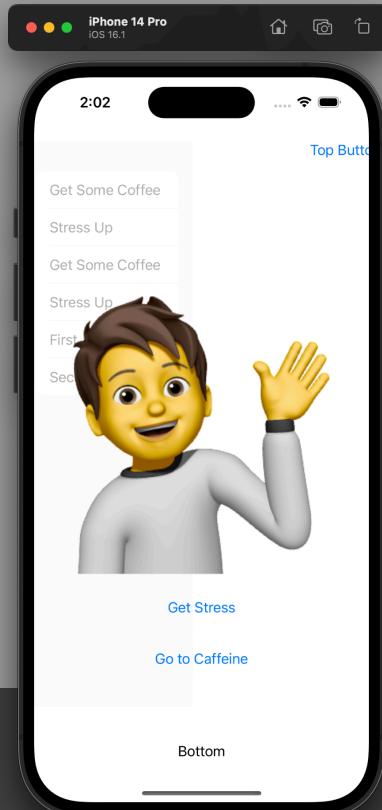
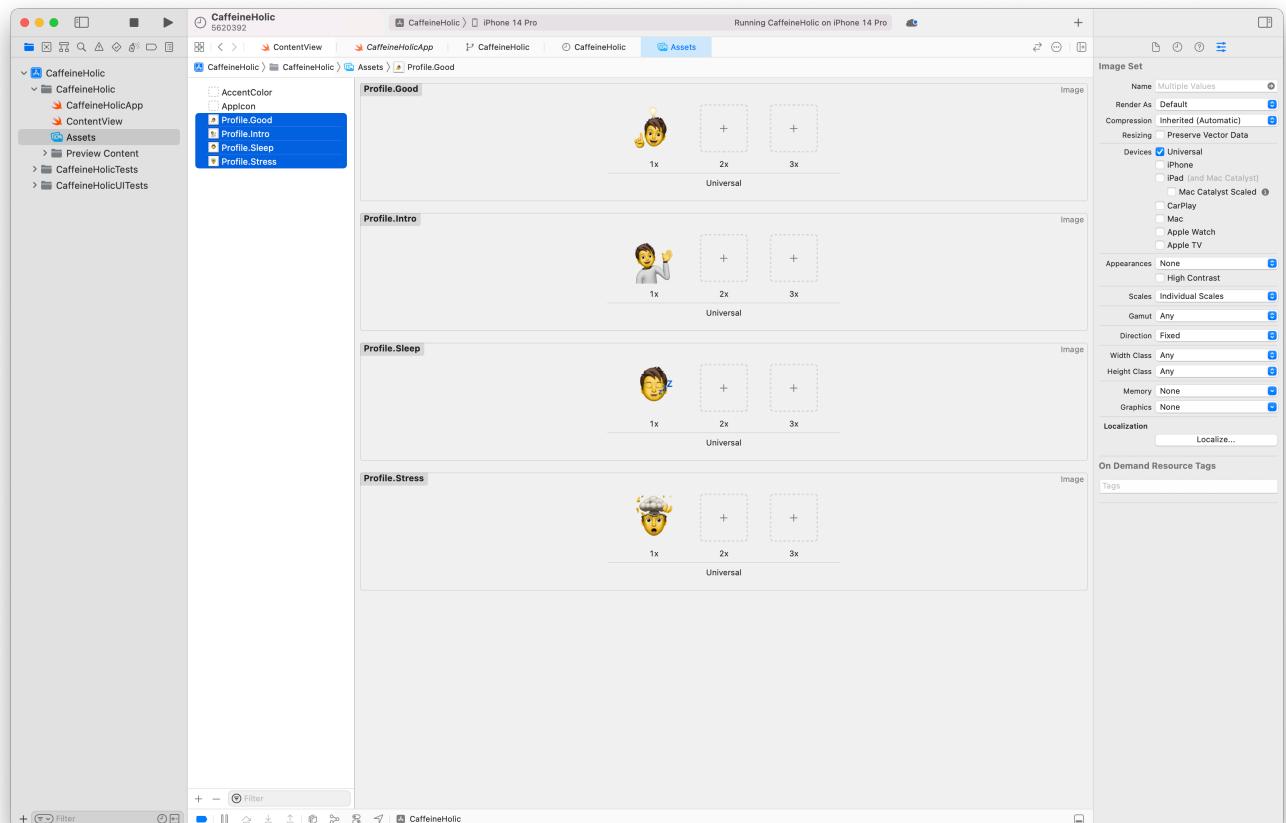
- (버튼 동작) 화면 중앙 버튼을 누르면 배경에 목록에 “Get Some Coffee”를 출력한다
- (화면 구성) “Go to Caffeine” 버튼을 추가한 뒤 배경 목록에 추가하도록 구현한다
- (화면 구성) Image를 추가하고, 원하는 이미지를 설정한다

미니 프로젝트 | DAY #01

# Step #2 버튼 동작 만들기

이미지  
추가

- Image 파일을 Assets에 추가함



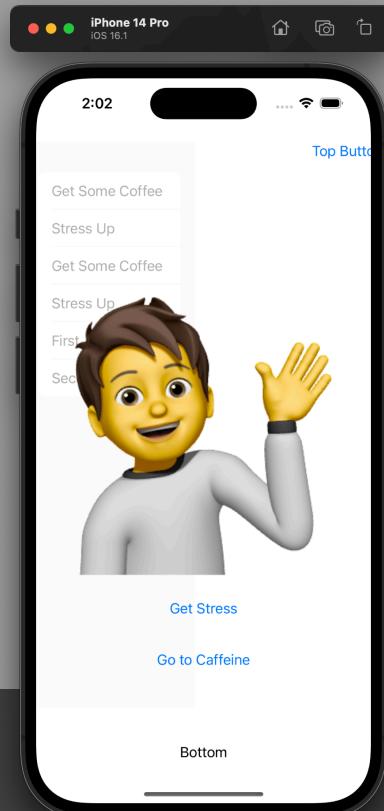
미니 프로젝트 | DAY #01

# Step #2 버튼 동작 만들기

## 버튼 동작

- (버튼 동작) 화면 중앙 버튼을 누르면 배경에 목록에 “Get Some Coffee”를 출력한다

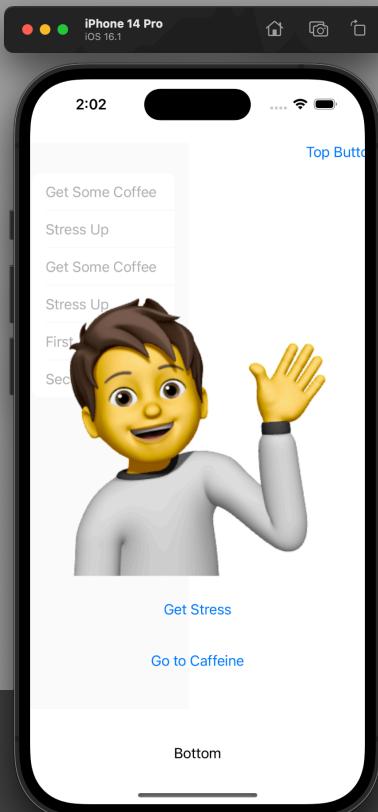
```
struct ContentView: View {  
    @State private var dailyList = ["First", "Second"]  
  
    func incList() {  
        dailyList.insert("New", at: 0)  
    }  
  
    func incCoffee() {  
        dailyList.insert("Get Some Coffee", at: 0)  
    }  
}
```



미니 프로젝트 | DAY #01

# Step #2 버튼 동작 만들기

## 버튼 동작



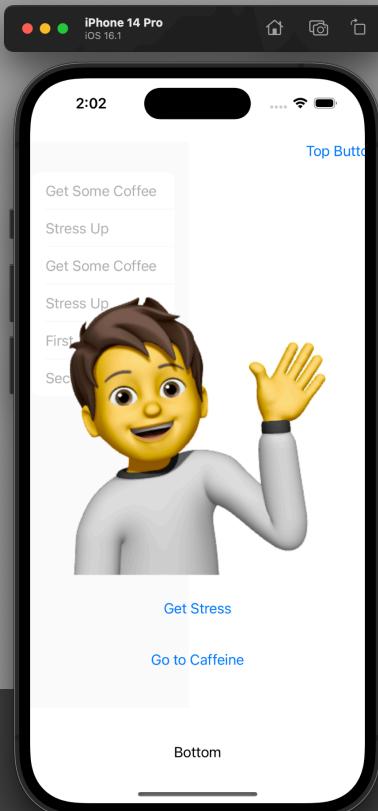
- (버튼 동작) 화면 중앙 버튼을 누르면 배경에 목록에 “Get Some Coffee”를 출력한다

```
53 var Buttons: some View {
54     VStack {
55         // Top buttons
56         HStack(spacing: 20){
57             Text("")
58             Spacer()
59         }
60         Button("Top Button") {
61             print("Top Button Click")
62         }
63     }
64     Spacer()
65
66
67     Button("Go to Caffeine") {
68         print("Click")
69         incList()
70         incCoffee()|
```

미니 프로젝트 | DAY #01

# Step #2 버튼 동작 만들기

## 버튼 동작



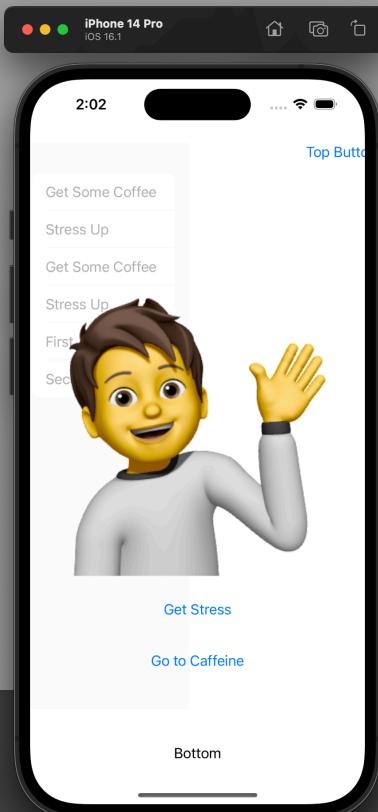
- (버튼 동작) 화면 중앙 버튼을 누르면 배경에 목록에 “Get Some Coffee”를 출력한다

```
53 var Buttons: some View {  
54     VStack {  
55         //  
56         Button("Go to Caffeine") {  
57             print("Click")  
58             incList()  
59             incCoffee()|  
60         }  
61         .padding()  
62     }  
63     //  
64     Button("Go to Caffeine") {  
65         print("Click")  
66         incList()  
67         incCoffee()|  
68     }  
69     .padding()  
70     Spacer()  
71     Text("Bottom")  
72 }  
73 .padding()  
74 }  
75 }  
76 }  
77 }  
78 }  
79 }  
80 }
```

미니 프로젝트 | DAY #01

# Step #2 버튼 동작 만들기

## 화면 구성



- (화면 구성) “Go to Caffeine” 버튼을 추가한 뒤 배경 목록에 추가하도록 구현한다

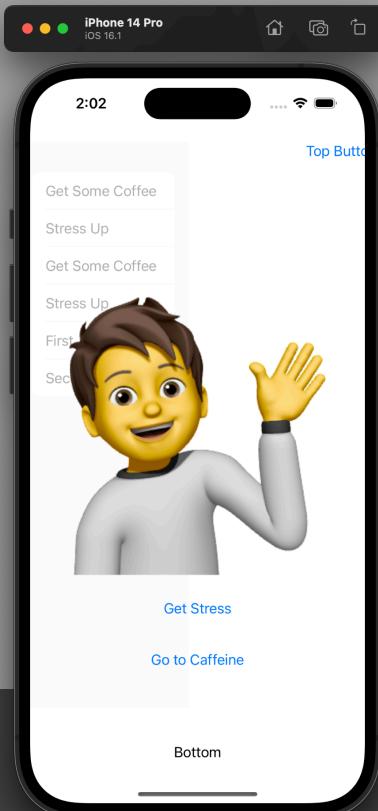
```
func incCoffee() {  
    dailyList.insert("Get Some Coffee", at: 0)  
}
```

```
func incStress() {  
    dailyList.insert("Stress Up", at: 0)  
}
```

```
func getRest() {  
    dailyList.insert("REST", at: 0)  
}
```

# Step #2 버튼 동작 만들기

## 화면 구성



- (화면 구성) “Go to Caffeine” 버튼을 추가한 뒤 배경 목록에 추가하도록 구현한다

```
var Buttons: some View {
    VStack {
        // Top buttons
        HStack(spacing: 20){
            Text("")
            Spacer()
        }
        / ...
        Text("Top Button")
        Button("Top Button") {
            print("Top Button Click")
        }
    }
    Spacer()
    Button("Get Stress") {
        print("")
        incStress()
    }
    .padding()

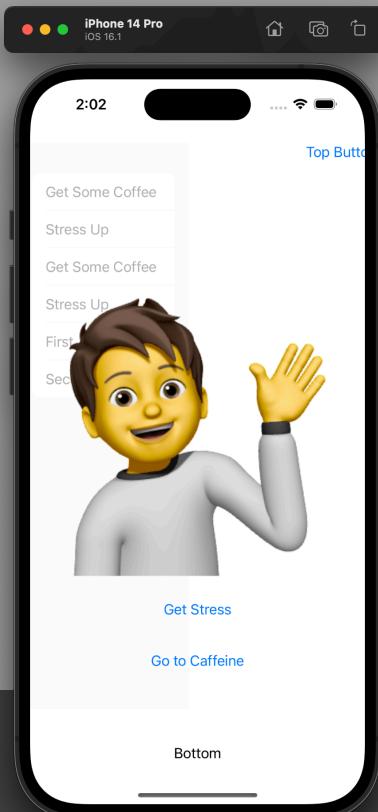
    Button("Go to Caffeine") {
        print("Click")
    }
}
```

미니 프로젝트

DAY #01

# Step #2 버튼 동작 만들기

## Image 구성



- (화면 구성) Image를 추가하고, 원하는 이미지를 설정한다

```
var Buttons: some View {
    VStack {
        // Top buttons
        HStack(spacing: 20){
            Text("")
            Spacer()
            Text("Top Button")
            Button("Top Button") {
                print("Top Button Click")
            }
        }
        Spacer()
        Image("Profile.Intro")
        Button("Get Stress") {
            print("")
            incStress()
        }
        .padding()
    }
}
```

미니 프로젝트 | DAY #01

# Step #2 버튼 동작 만들기

## Image 구성

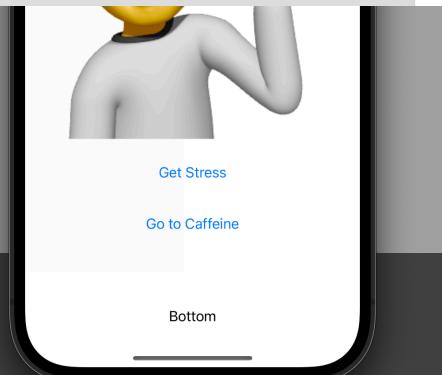
```
CaffeineHolic
└── CaffeineHolic
    ├── CaffeineHolicApp
    └── ContentView
```

M

```
Assets
└── Preview Content
```

```
CaffeineHolicTests
```

```
CaffeineHolicUITests
```



- (화면 구성) Image를 추가하고, 원하는 이미지를 설정한다

```
var Buttons: some View {
    VStack {
        // Top buttons
    }
}
```

```
AccentColor
AppIcon
Profile.Good
Profile.Intro
Profile.Sleep
Profile.Stress
```

```
Profile.Intro
```

```
1x 2x 3x
```

```
Universal
```

```
print("")
incStress()
}

.padding()
```

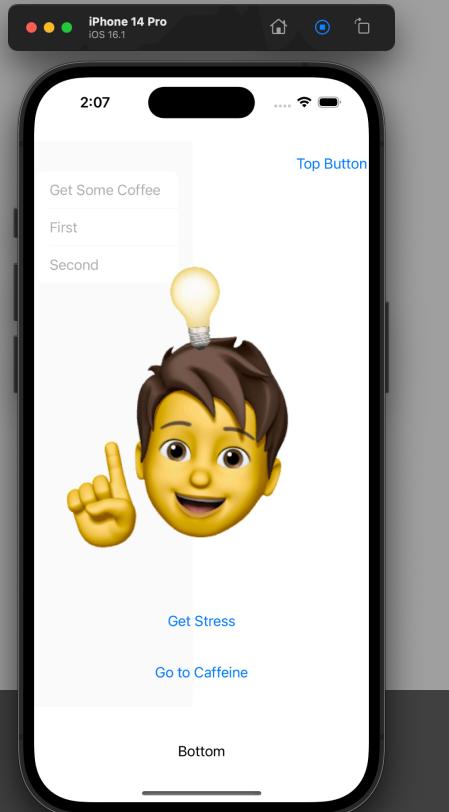
미니 프로젝트 | DAY #01

# Step #3

카페인 상태 모델 만들기

## Step #3 목표

- (코드 분리) 카페인 상태 모델 작성을 위해 “CaffeineModel.swift” 파일 생성 및 작성
- (화면 구성) 기존 화면과 연결된 코드를 상태 모델에 연결
- (화면 구성) 각 상태에 이미지 연결 코드 추가



# Step #3 카페인 상태 모델 만들기

코드  
분리

- (코드 분리) 카페인 상태 모델 작성을 위해  
“CaffeineModel.swift” 파일 생성 및 작성

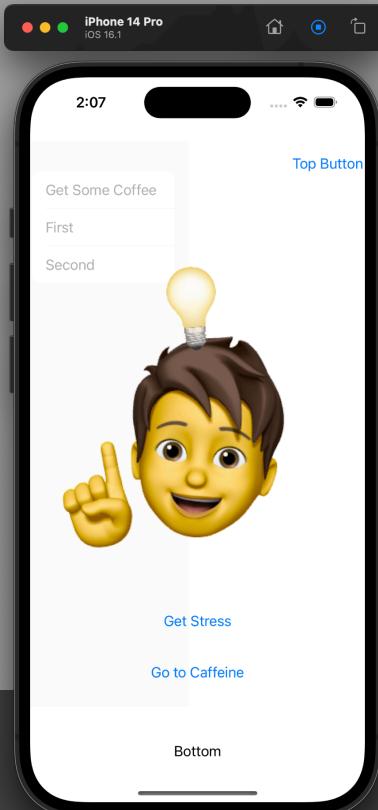
```
9  
10 class CaffeineModel {  
11     enum CaffeineState {  
12         case Intro  
13         case wakening  
14         case stressful  
15         case rest  
16     }  
17 }
```



미니 프로젝트 | DAY #01

# Step #3 카페인 상태 모델 만들기

## 상태 모델 연결



- (화면 구성) 기존 화면과 연결된 코드를 상태 모델에 연결

```
10 class CaffeineModel {  
11     enum CaffeineState {  
12         case Intro  
13         case wakening  
14         case stressful  
15         case rest  
16     }  
17  
18     var currentState:CaffeineState = CaffeineState.Intro  
19  
20     func getStateImg () -> String {  
21         switch(currentState) {  
22             case .Intro:  
23                 return "Profile.Intro"  
24             default:  
25                 return "Profile.Intro"  
26         }  
27     }  
28  
29 }
```

# Step #3 카페인 상태 모델 만들기

## 상태 모델 연결

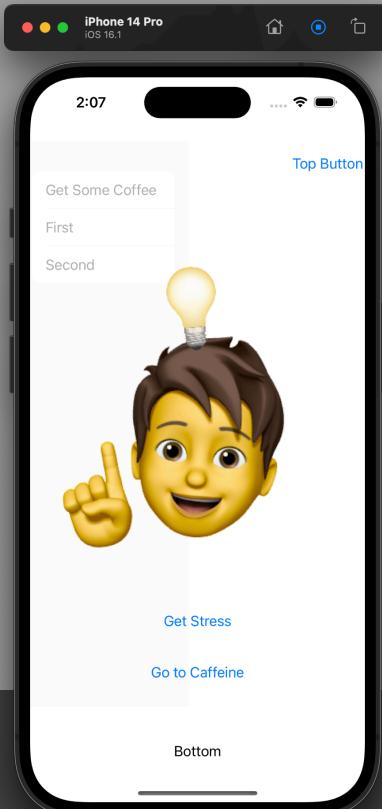


- (화면 구성) 기존 화면과 연결된 코드를 상태 모델에 연결

```
14
15 struct ContentView: View {
16     @State private var dailyList = ["First", "Second"]
17     var caffeine:CaffeineModel = CaffeineModel()
18
19     func incList() {
20         dailyList.insert("New", at: 0)
21     }
22 }
```

# Step #3 카페인 상태 모델 만들기

## 상태 모델 연결



- (화면 구성) 기존 화면과 연결된 코드를 상태 모델에 연결

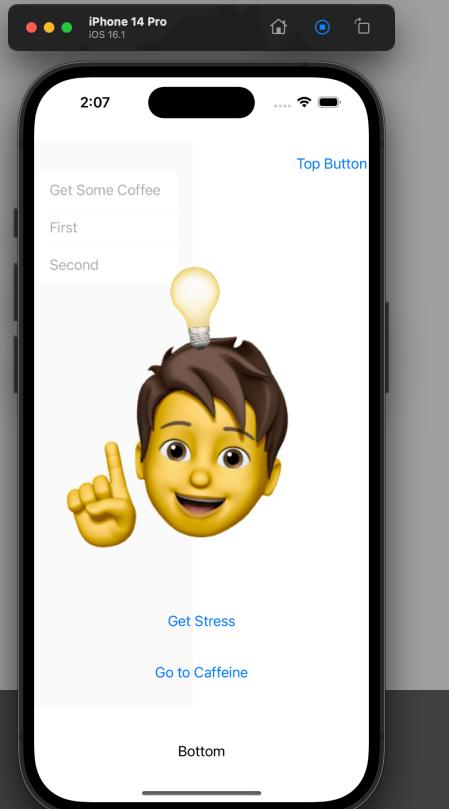
```
var Buttons: some View {
    VStack {
        // Top buttons
        HStack(spacing: 20){
            Text("")
            Spacer()
            ...
            .padding()
            Spacer()
            Image(caffeine.getStateImg())
            Button("Get Stress") {
                print("")
                incStress()
            }
            .padding()
        }
    }
}
```

# Step #3 카페인 상태 모델 만들기

## 이미지 연결

- (화면 구성) 각 상태에 이미지 연결 코드 추가

```
class CaffeineModel {  
  
    func getStateImg () -> String {  
        switch(currentState) {  
            case .Intro:  
                return "Profile.Intro"  
            case .rest:  
                return "Profile.Rest"  
            case .stressful:  
                return "Profile.Stressful"  
            case .wakening:  
                return "Profile.Wakening"  
        }  
    }  
  
    func doWakening() {  
        self.currentState = CaffeineState.wakening  
    }  
}
```



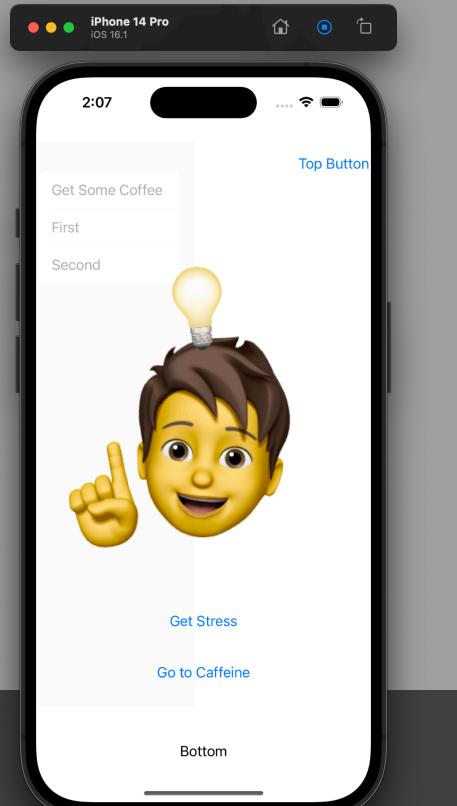
# Step #3 카페인 상태 모델 만들기

## 이미지 연결

- (화면 구성) 각 상태에 이미지 연결 코드 추가

```
struct ContentView: View {  
    @State private var dailyList = ["First", "Second"]  
    @State var caffeine:CaffeineModel = CaffeineModel()
```

```
func incCoffee() {  
    dailyList.insert("Get Some Coffee", at: 0)  
    caffeine.doWakening()  
}
```

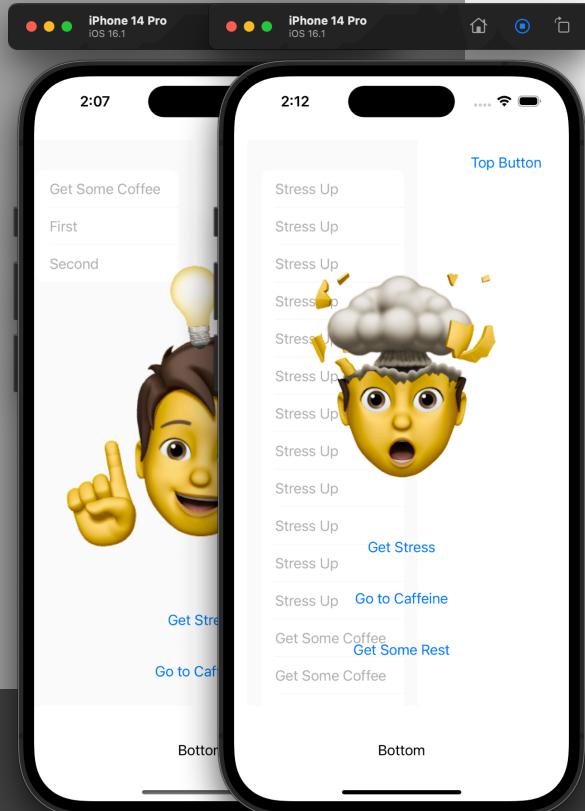


# Step #4

카페인 상태 모델 심화 작성

## Step #4 목표

- 모든 상태로 이동하는 코드 작성
- 현재 상태와 같은 상태를 누를 경우, 점점 크기를 증가함
- 초기 상태로 돌아갈 수 있도록 함



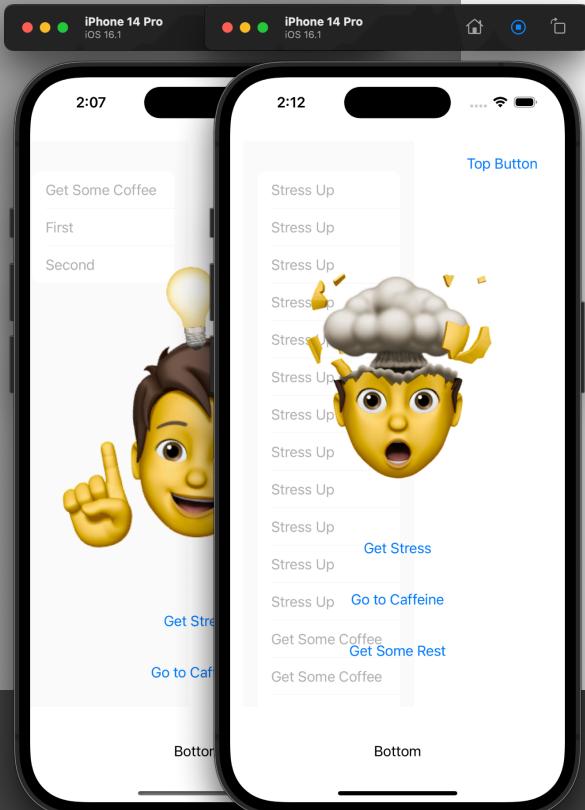
# Step #4 카페인 상태 모델 심화 작성

상태이전  
코드

- 모든 상태로 이동하는 코드 작성

```
class CaffeineModel {  
    enum CaffeineState: String {  
        case Intro  
        case Wakening  
        case Stressful  
        case Rest  
    }  
}
```

```
func getStateImg () -> String {  
    return "Profile." + currentState.rawValue  
}
```



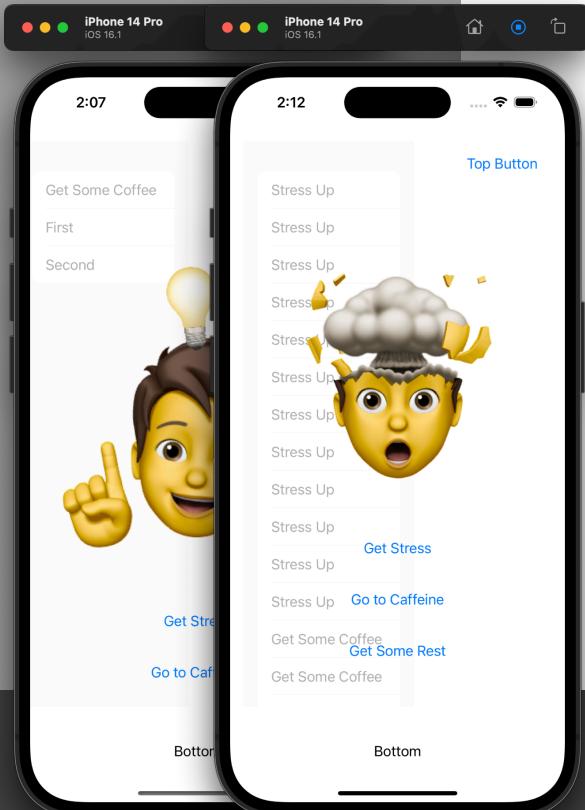
미니 프로젝트

DAY #01

# Step #4 카페인 상태 모델 심화 작성

상태이전  
코드

- 모든 상태로 이동하는 코드 작성



```
func doWakening(){
    self.currentState = CaffeineState.Wakening
}

func doStress(){
    self.currentState = CaffeineState.Stressful
}

func doRest(){
    self.currentState = CaffeineState.Rest
}
```

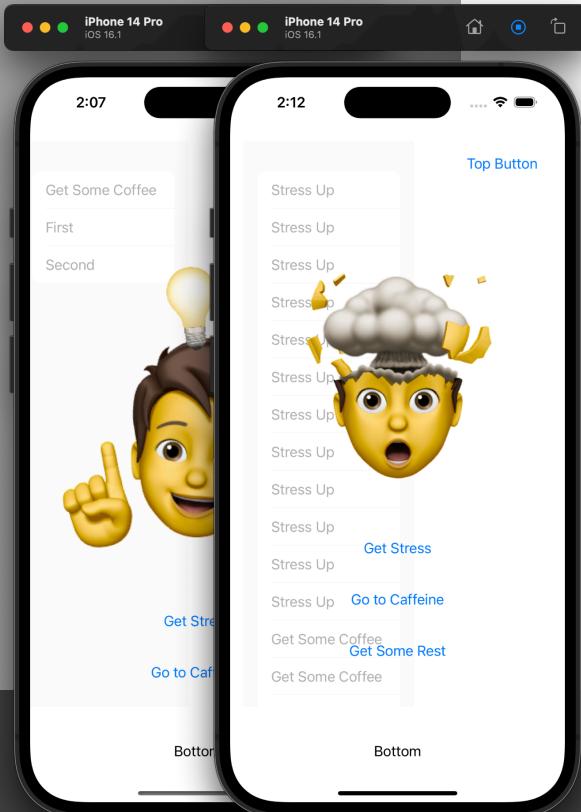
미니 프로젝트 | DAY #01

# Step #4 카페인 상태 모델 심화 작성

상태이전  
코드

- 모든 상태로 이동하는 코드 작성

```
var Buttons: some View {  
    VStack {  
  
        Button("Get Stress") {  
            print("")  
            incStress()  
        }  
        .padding()  
  
        Button("Go to Caffeine") {  
            print("Click")  
            incCoffee()  
        }  
        .padding()  
  
        Button("Get Some Rest") {  
            getRest()  
        }.padding()  
  
        Spacer()  
  
        Text("Bottom")  
    }  
}
```



미니 프로젝트

DAY #01

# Step #4 카페인 상태 모델 심화 작성

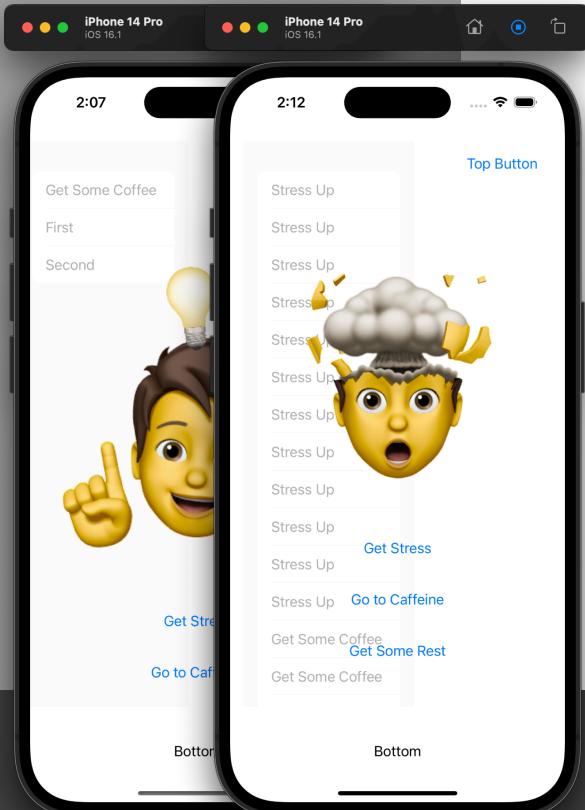
## 상태이전 코드

- 모든 상태로 이동하는 코드 작성

```
func incStress() {  
    dailyList.insert("Stress Up", at: 0)  
    caffeine.doStress() ⚠️ Result of call to 'd  
}
```

```
func incCoffee() {  
    dailyList.insert("Get Some Coffee", at: 0)  
    caffeine.doWakening() ⚠️ Result of call to 'doW  
}
```

```
func getRest() {  
    dailyList.insert("REST", at: 0)  
    caffeine.doRest() ⚠️ Result of call to 'd  
}
```

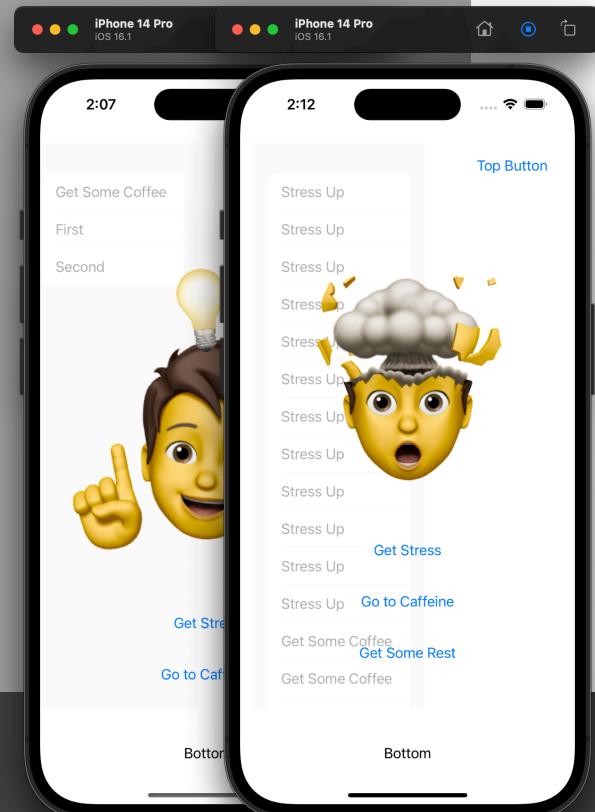


# Step #4 카페인 상태 모델 심화 작성

## 크기변경

- 현재 상태와 같은 상태를 누를 경우, 점점 크기를 증가함

```
class CaffeineModel {  
  
    var currentState:CaffeineState = CaffeineState.Intro  
    var imgFrame:CGSize = CGSize(width: 200.0, height: 200.0)
```



```
func incImgSize() {  
    var newFrame:CGSize = imgFrame  
    newFrame.height += 10  
    newFrame.width += 10  
    imgFrame = newFrame  
}
```

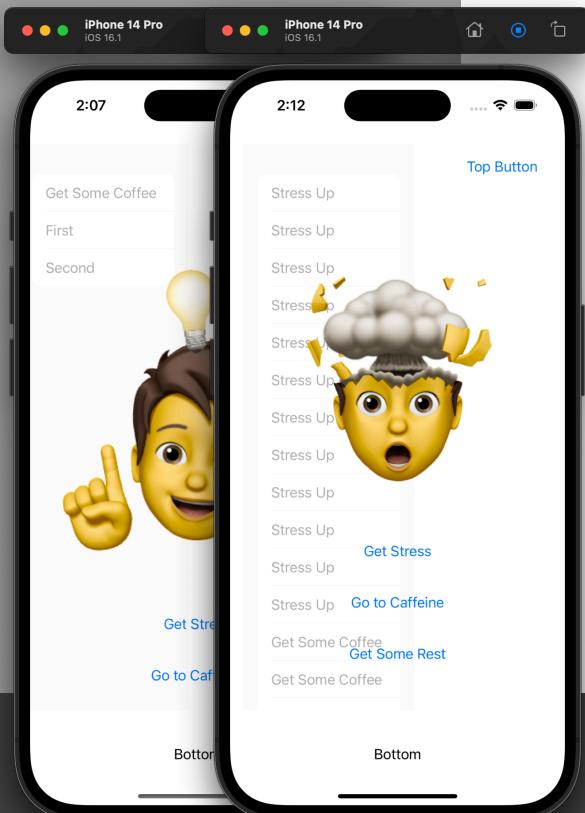
# Step #4 카페인 상태 모델 심화 작성

## 크기변경

- 현재 상태와 같은 상태를 누를 경우, 점점 크기를 증가함

```
var Buttons: some View {
    VStack {
        Spacer()
        Image(caffeine.getStateImg())
            .resizable()
            .frame(width: caffeine.imgFrame.width, height:
                caffeine.imgFrame.height)
    }
}

Button("Get Stress") {
```



# Step #4 카페인 상태 모델 심화 작성

## 크기변경

- 현재 상태와 같은 상태를 누를 경우, 점점 크기를 증가함

```
private func changeState(newState:CaffeineState) -> (Bool, [String]) {
    var result:Bool = false

    if newState != self.currentState {
        result = true
        self.currentState = newState
        resetImgSize()
    }

    // Bigger Img Size
    else {
        incImgSize()
    }

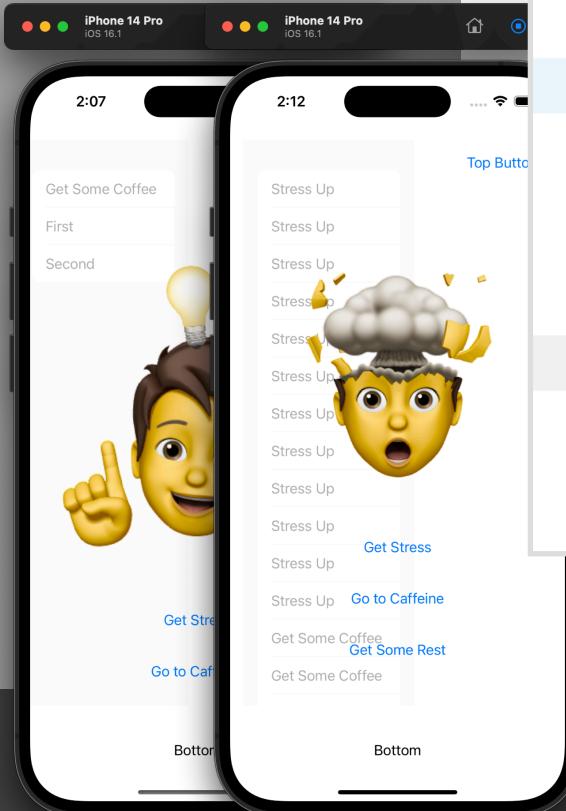
    return (result, dailyList)
}
```



# Step #4 카페인 상태 모델 심화 작성

## 크기변경

- 현재 상태와 같은 상태를 누를 경우, 점점 크기를 증가함



```
func doWakening(){
    changeState(newState: .Wakened)
}

func doStress(){
    changeState(newState: .Stressful)
}

func doRest(){
    changeState(newState: .Rest)
}
```

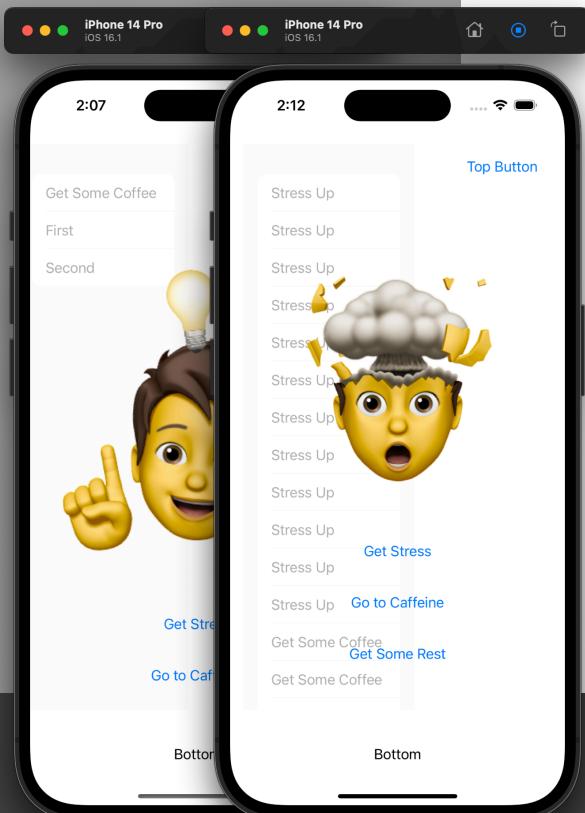
# Step #4 카페인 상태 모델 심화 작성

## 모델심화

- 데일리 리스트 관리 기능 통합

```
class CaffeineModel {
```

```
    let originList = ["First", "Second"]  
    var dailyList:[String] = []
```



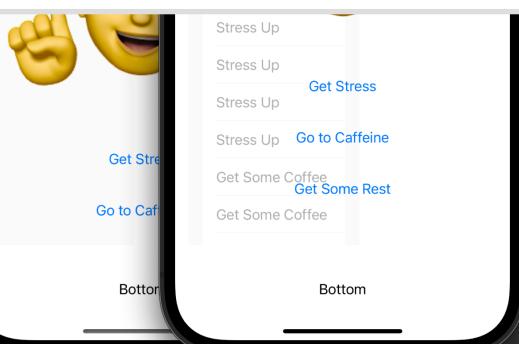
# Step #4 카페인 상태 모델 심화 작성

## 모델심화

- 데일리 리스트 관리 기능 통합

```
class CaffeineModel {
```

```
    • private func addDailyList(state:CaffeineState) {  
        switch(state) {  
            case .Stressful: dailyList.insert("Stress Up", at: 0);  
            case .Wakening: dailyList.insert("Get Some Coffee", at: 0);  
            case .Intro: dailyList.insert("New", at: 0);  
            case .Rest: dailyList.insert("REST", at: 0);  
        }  
    }
```



미니 프로젝트 | DAY #01

# Step #4 카페인 상태 모델 심화 작성

## 모델심화

- 데일리 리스트 관리 기능 통합

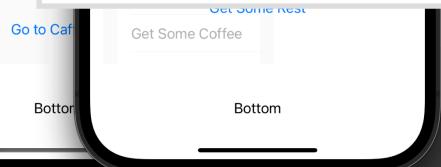
```
private func changeState(newState:CaffeineState) -> (Bool, [String]) {
    var result:Bool = false

    addDailyList(state: newState)

    if newState != self.currentState {
        result = true
        self.currentState = newState
    }

    // Bigger Img Size
    else {
        incImgSize()
    }

    return (result, dailyList)
}
```



# Step #4 카페인 상태 모델 심화 작성

## 모델심화

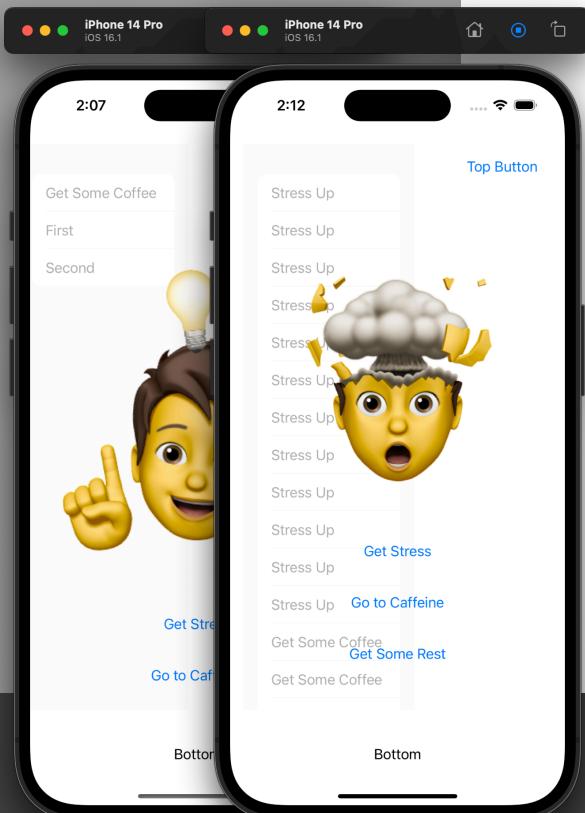
- 데일리 리스트 관리 기능 통합

```
class CaffeineModel {
```

```
    func doWakening() -> (Bool, [String]) {
        return changeState(newState: .Wakening)
    }

    func doStress() ->(Bool, [String]){
        return changeState(newState: .Stressful)
    }

    func doRest() ->(Bool, [String]){
        return changeState(newState: .Rest)
    }
```



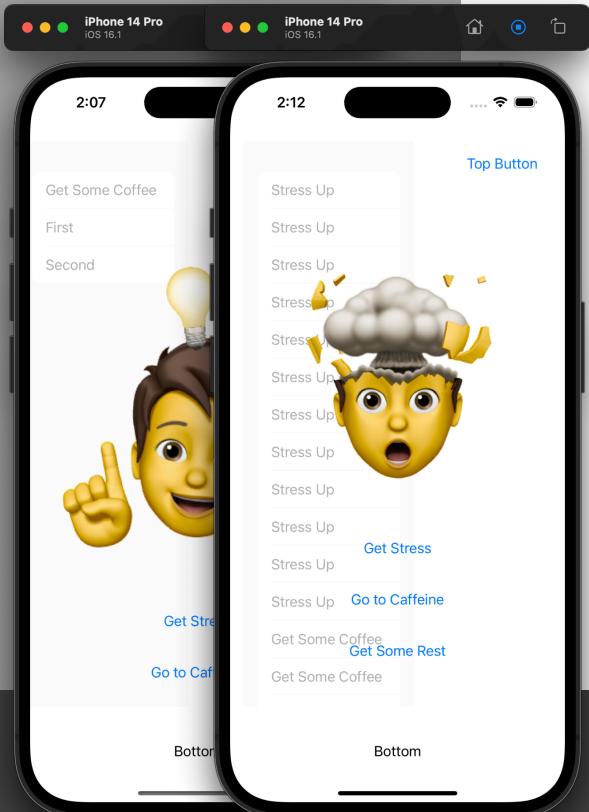
미니 프로젝트 | DAY #01

# Step #4 카페인 상태 모델 심화 작성

초기상태  
복원

- 초기 상태로 돌아가는 코드 작성

```
class CaffeineModel {  
    ...  
  
    func incImgSize() {  
        var newFrame:CGSize = imgFrame  
        newFrame.height += 10  
        newFrame.width += 10  
        if newFrame.height < 350 {  
            imgFrame = newFrame  
        }  
    }  
  
    func resetImgSize() {  
        imgFrame = CGSize(width: 200.0, height: 200.0)  
    }  
}
```



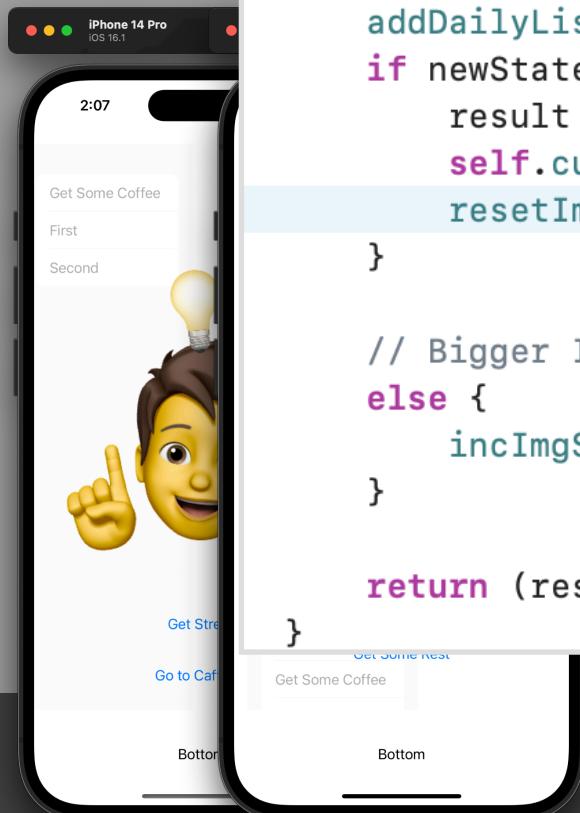
미니 프로젝트 | DAY #01

# Step #4 카페인 상태 모델 심화 작성

초기  
복원

- 초기 상태로 돌아가는 코드 작성

```
class CaffeineModel {  
  
    private func changeState(newState:CaffeineState) -> (Bool, [String]) {  
        var result:Bool = false  
  
        addDailyList(state: newState)  
        if newState != self.currentState {  
            result = true  
            self.currentState = newState  
            resetImgSize()  
        }  
  
        // Bigger Img Size  
        else {  
            incImgSize()  
        }  
  
        return (result, dailyList)  
    }  
}
```

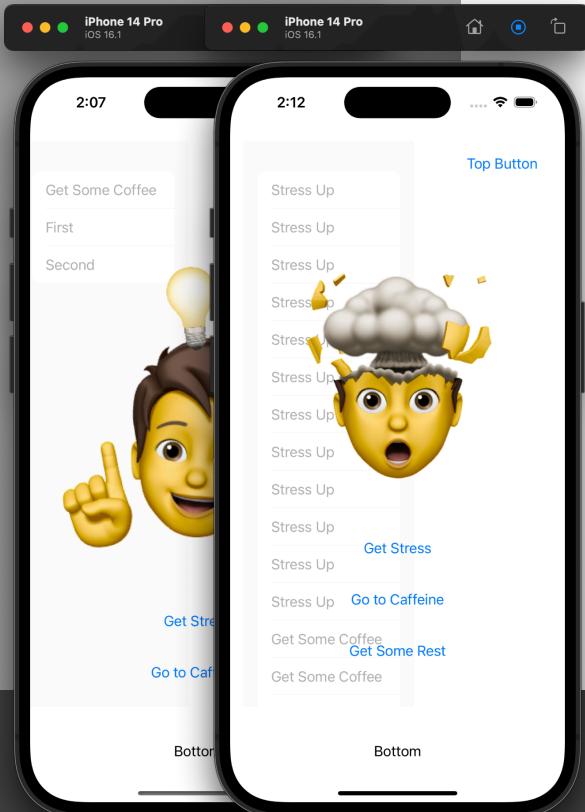


# Step #4 카페인 상태 모델 심화 작성

초기상태  
복원

- 초기 상태로 돌아가는 코드 작성

```
var Buttons: some View {
    VStack {
        // Top buttons
        HStack(spacing: 20){
            Text("")
            Spacer()
            Text("Top Button")
            Button("Top Button") {
                print("Top Button Click")
                //resetState()
                (result, dailyList) = caffeine.doReset()
            }
        }
        .padding()
    }
}
```

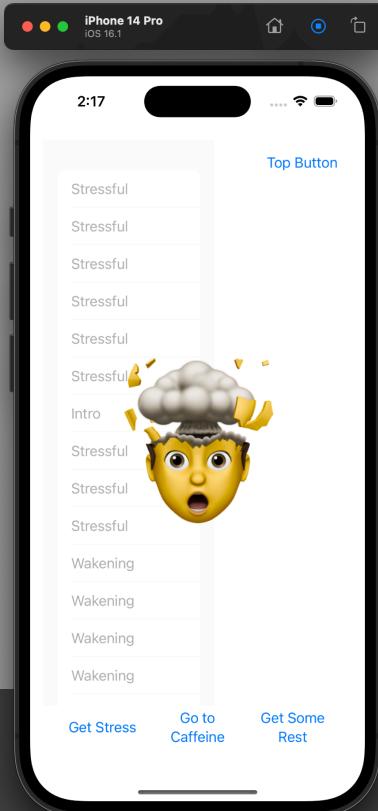


미니 프로젝트 | DAY #01

# Step #Extra#

자잘한 여러가지 수정

## Step #@ 목표



- 열거형의 여러가지 사용법
  - String Value 사용법
- 버튼 하단 이동
- 배경 목록에서 중복된 상태가 나열되지 않도록  
“상태 + #n”의 숫자 누적 형식으로 표현 변경하기

미니 프로젝트 | DAY #01