Студенческая
ИТ-лаборатория

Архитектура SwiftUI навигации

Виктор Глебов
iOS Developer at Effective

Базовая навигация

Базовая навигация

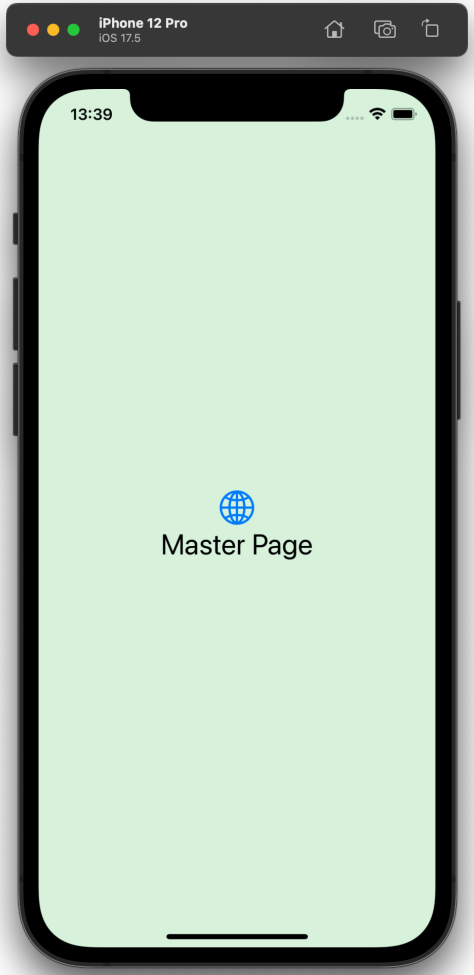
effectiveband

```
struct BasicViewConfiguration: Identifiable {
    let id: UUID
    let title: String
    let backgroundColor: Color
    let imageName: String

    public init(
        title: String = "Master",
        backgroundColor: Color = .green,
        imageName: String = "globe"
    ) {
        self.id = UUID()
        self.title = title
        self.backgroundColor = backgroundColor
        self.imageName = imageName
    }
}
```

```
struct PlaceHolderView: View {
    let configuration: BasicViewConfiguration

    public init(configuration: BasicViewConfiguration) {
        self.configuration = configuration
    }

    var body: some View {
        VStack {
            Image(systemName: configuration.imageName)
                .font(.largeTitle)
                .foregroundStyle(.tint)
            Text("\(configuration.title) Page")
                .font(.title)
        }
        .frame(maxWidth: .infinity, maxHeight: .infinity)
        .background(configuration.backgroundColor.opacity(0.2))
        .navigationBarTitle(configuration.title)
    }
}
```

Базовая навигация

```swift
import SwiftUI

@main
struct native_navigationApp: App {
    var body: some Scene {
        WindowGroup {
            MasterView()
        }
    }
}

#Preview {
    MasterView()
}
```



effectiveband

Студенческая
ИТ-лаборатория

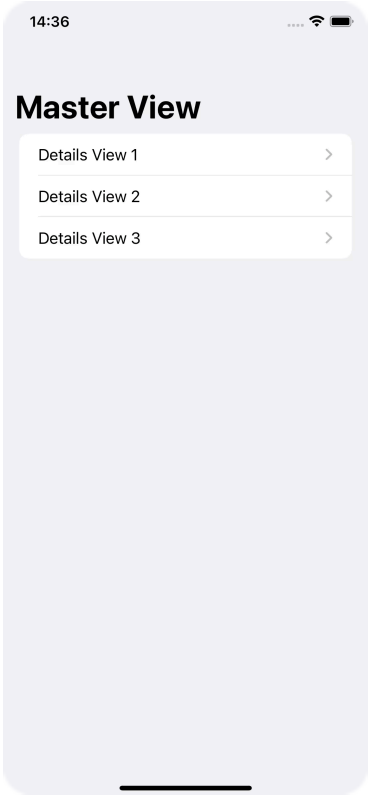
effectiveband

NavigationView (SwiftUI 1.0 / iOS 13+)

NavigationView (basic scenario)

effectiveband

```
struct MasterView: View {
    let testData: [BasicViewConfiguration] = [
        BasicViewConfiguration(
            title: "Details View 1", backgroundColor: .purple, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 2", backgroundColor: .orange, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 3", backgroundColor: .teal, imageName: "scribble.variable"
        )
    ]

    var body: some View {
        NavigationView {
            List(testData, id: \.id) { item in
                NavigationLink(destination: PlaceHolderView(configuration: item)) {
                    Text(item.title)
                }
            }
            .navigationBarTitle("Master View")
        }
    }
}
```



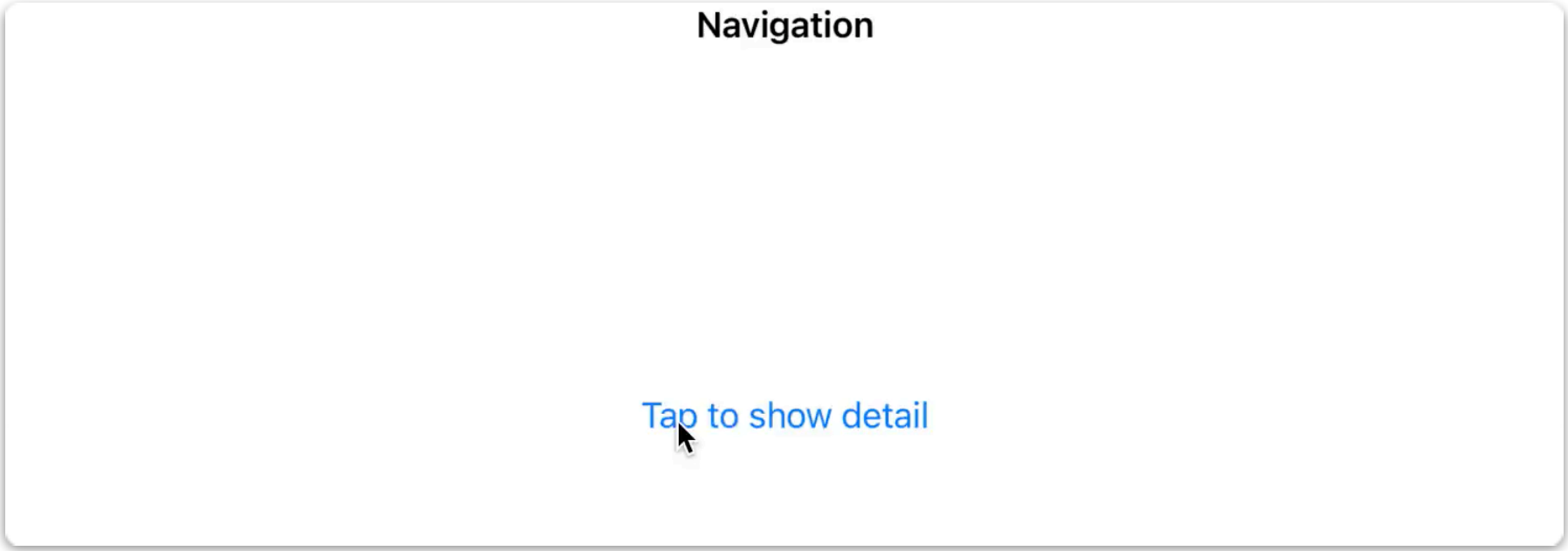
NavigationView (programmatic)

effectiveband

```
struct ContentView: View {
    @State private var isShowingDetailView = false

    var body: some View {
        NavigationView {
            VStack {
                NavigationLink(destination: Text("Second View"), isActive: $isShowingDetailView) { EmptyView() }

                Button("Tap to show detail") {
                    isShowingDetailView = true
                }
            }
            .navigationTitle("Navigation")
        }
    }
}
```

NavigationView (programmatic)

effectiveband



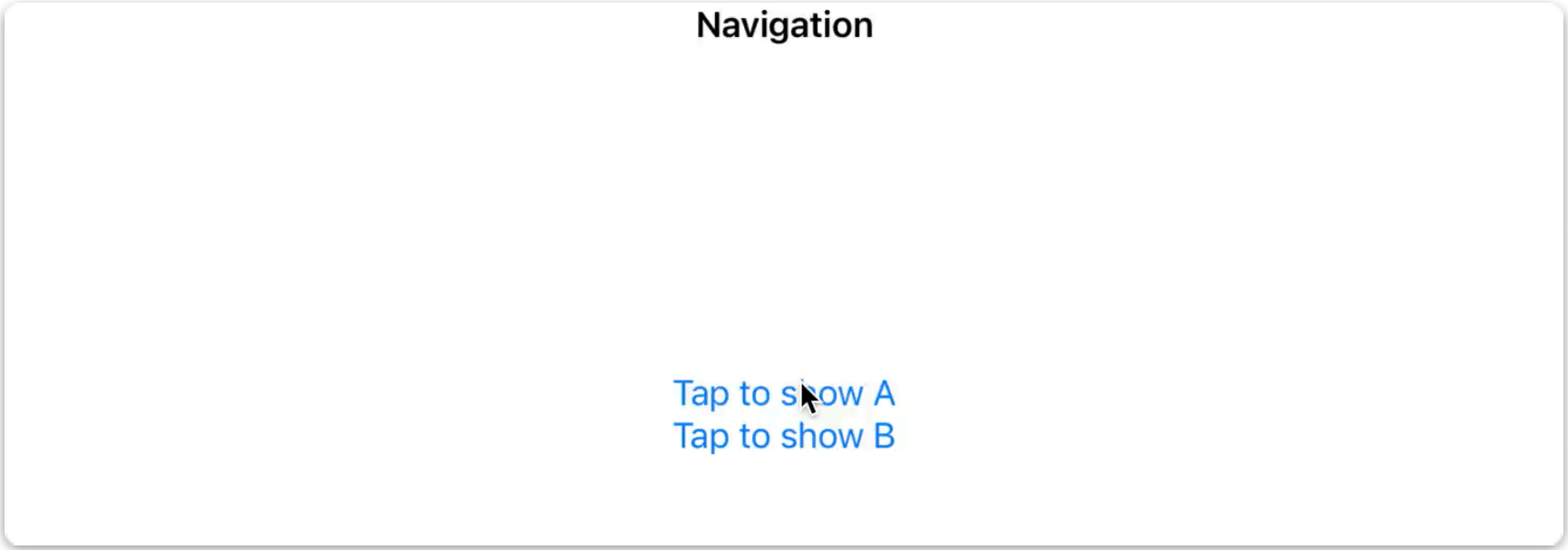
NavigationView (programmatic-scalable)

effectiveband

```
struct ContentView: View {
    @State private var selection: String? = nil

    var body: some View {
        NavigationView {
            VStack {
                NavigationLink(destination: Text("View A"), tag: "A", selection: $selection) { EmptyView() }
                NavigationLink(destination: Text("View B"), tag: "B", selection: $selection) { EmptyView() }

                Button("Tap to show A") {
                    selection = "A"
                }

                Button("Tap to show B") {
                    selection = "B"
                }
            }
            .navigationTitle("Navigation")
        }
    }
}
```

NavigationView (programmatic-scalable)

effectiveband



NavigationView (cons)

effectiveband

- **Ограниченный доступ к стеку ViewController**
  - Нет обратной навигации (pop)
  - Нет перехода на Root экран (reset)
- **Ограниченная программная навигация**
  - Нет состояния навигации
  - Нет истории навигации
- **Deprecated (Since WWDC2022, iOS 16+)**

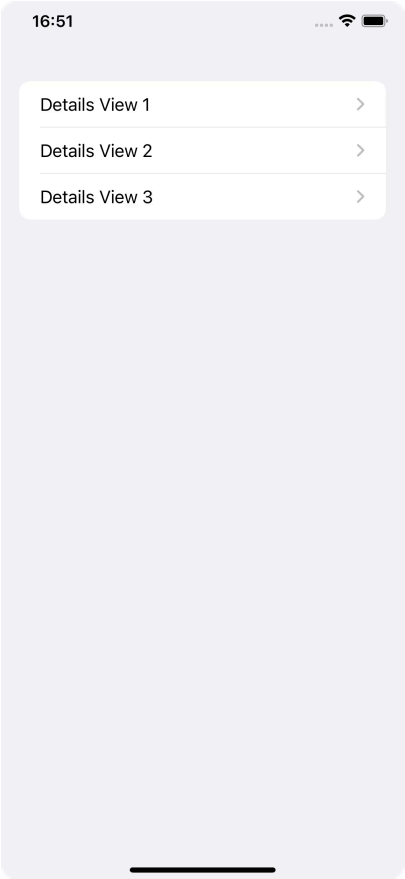
Студенческая
ИТ-лаборатория

effectiveband

NavigationStack (WWDC2022 / iOS 16+)

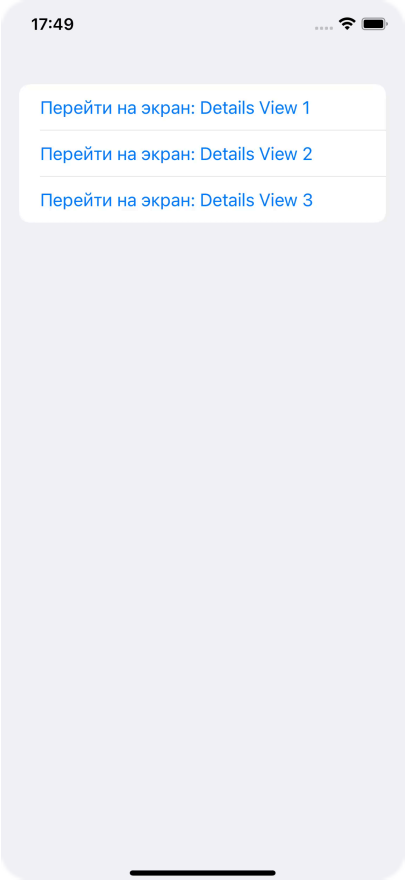
NavigationStack (basic scenario)

```
struct MasterView: View {
    let testData: [BasicViewConfiguration] = [
        BasicViewConfiguration(
            title: "Details View 1", backgroundColor: .purple, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 2", backgroundColor: .orange, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 3", backgroundColor: .teal, imageName: "scribble.variable"
        )
    ]

    var body: some View {
        NavigationStack {
            List(testData, id: \.id) { item in
                NavigationLink(destination: PlaceHolderView(configuration: item)) {
                    Text(item.title)
                }
            }
        }
    }
}
```



NavigationStack (programmatic)

```
struct MasterView: View {
    @State private var path: [BasicViewConfiguration] = []

    let testData: [BasicViewConfiguration] = [
        BasicViewConfiguration(
            title: "Details View 1", backgroundColor: .purple, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 2", backgroundColor: .orange, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 3", backgroundColor: .teal, imageName: "scribble.variable"
        )
    ]

    var body: some View {
        NavigationStack(path: $path) {
            List(testData, id: \.id) { item in
                Button("Перейти на экран: \(item.title)") {
                    path.append(item)
                }
            }
            .navigationDestination(for: BasicViewConfiguration.self) { destination in
                PlaceHolderView(configuration: destination)
            }
        }
    }
}
```



NavigationStack (back)

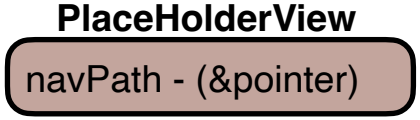
effectiveband

```
struct PlaceHolderView: View {
    let configuration: BasicViewConfiguration
    @Binding var navPath: [BasicViewConfiguration] // New

    public init(
        configuration: BasicViewConfiguration,
        navPath: Binding<[BasicViewConfiguration]> = .constant([]) // New
    ) {
        self.configuration = configuration
        self._navPath = navPath // New
    }

    var body: some View {
        VStack {
            Image(systemName: configuration.imageName)
                .font(.largeTitle)
                .foregroundStyle(.tint)
            Text("\(configuration.title) Page")
                .font(.title)
            Button("Back to Previous Screen") { // New
                if !navPath.isEmpty {
                    navPath.removeLast()
                }
            }
        }
        .frame(maxWidth: .infinity, maxHeight: .infinity)
        .background(configuration.backgroundColor.opacity(0.2))
        .navigationBarTitle(configuration.title)
    }
}
```

```
struct MasterView: View {
    @State private var path: [BasicViewConfiguration] = []

    let testData: [BasicViewConfiguration] = [
        BasicViewConfiguration(
            title: "Details View 1", backgroundColor: .purple, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 2", backgroundColor: .orange, imageName: "scribble.variable"
        ),
        BasicViewConfiguration(
            title: "Details View 3", backgroundColor: .teal, imageName: "scribble.variable"
        )
    ]

    var body: some View {
        NavigationStack(path: $path) {
            List(testData, id: \.id) { item in
                Button("Перейти на экран: \(item.title)") {
                    path.append(item)
                }
            }
            .navigationDestination(for: BasicViewConfiguration.self) { destination in
                PlaceHolderView(configuration: destination, navPath: $path) // New
            }
        }
    }
}
```

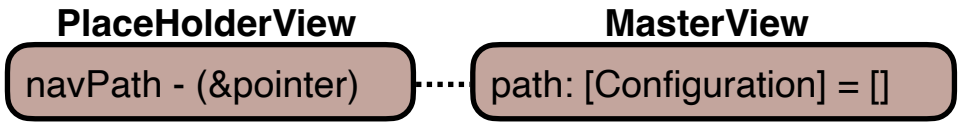
NavigationStack (back)

effectiveband



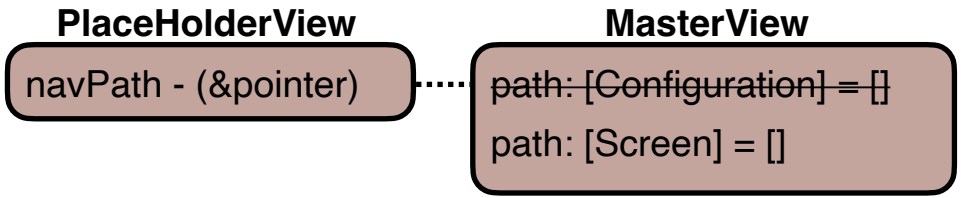
NavigationStack (back)

effectiveband



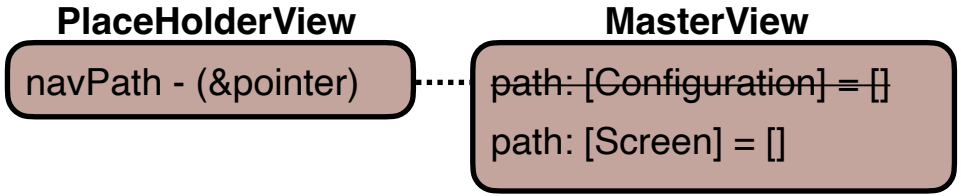
NavigationStack (back)

effectiveband



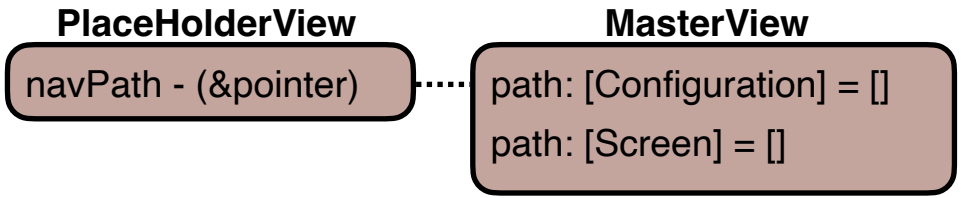
NavigationStack (back)

effectiveband

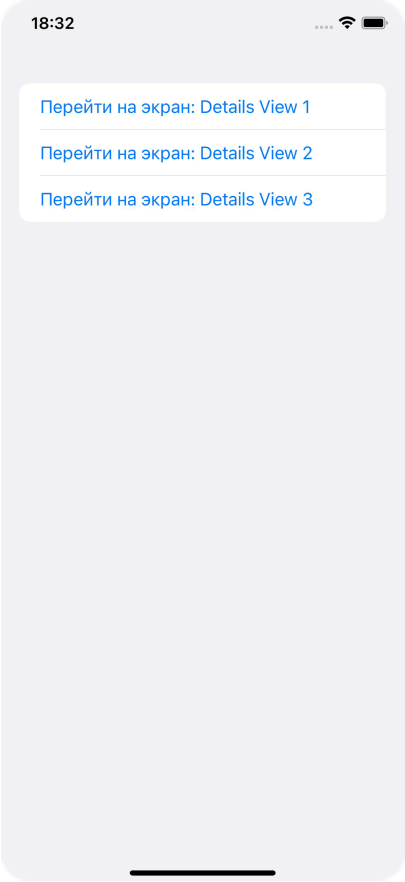



```
.navigationDestination(for: BasicViewConfiguration.self)
   {  destination in
         PlaceHolderView(configuration: destination, navPath: $path)
   }
```

NavigationStack (back)

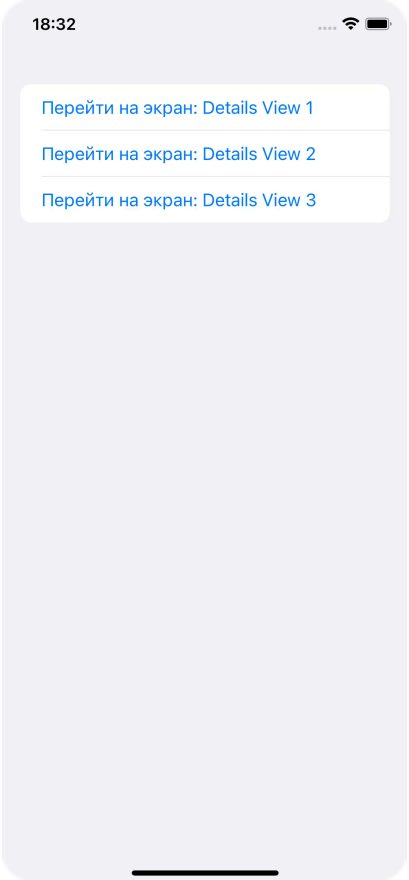



```
.navigationDestination(for: BasicViewConfiguration.self)
   {  destination in
         PlaceHolderView(configuration: destination, navPath: $path)
   }
```



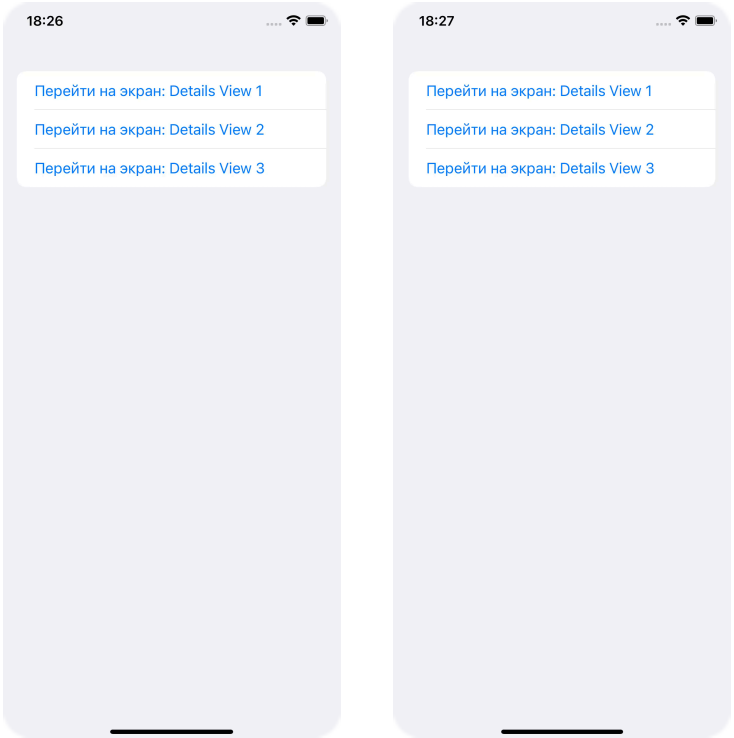
NavigationStack (dismiss)

```
struct PlaceHolderView: View {
    let configuration: BasicViewConfiguration
    @Binding var navPath: [BasicViewConfiguration]
    @Environment(\.dismiss) private var dismiss

    public init(
        configuration: BasicViewConfiguration,
        navPath: Binding<[BasicViewConfiguration]> = .constant([])
    ) {
        self.configuration = configuration
        self._navPath = navPath
    }

    var body: some View {
        VStack {
            Image(systemName: configuration.imageName)
                .font(.largeTitle)
                .foregroundStyle(.tint)
            Text("\(configuration.title) Page")
                .font(.title)
            Button("Back to Previous Screen") { // New
                dismiss()
            }
        }
        .frame(maxWidth: .infinity, maxHeight: .infinity)
        .background(configuration.backgroundColor.opacity(0.2))
        .navigationBarTitle(configuration.title)
    }
}
```



effectiveband

NavigationStack (reset)




```swift
struct PlaceHolderView: View {
    let configuration: BasicViewConfiguration
    @Binding var navPath: [BasicViewConfiguration]

    public init(
        configuration: BasicViewConfiguration,
        navPath: Binding<[BasicViewConfiguration]> = .constant([])
    ) {
        self.configuration = configuration
        self._navPath = navPath
    }

    var body: some View {
        VStack {
            Image(systemName: configuration.imageName)
                .font(.largeTitle)
                .foregroundStyle(.tint)
            Text("\(configuration.title) Page")
                .font(.title)
            Button("Go to Third Screen") { // New
                navPath.append(
                    BasicViewConfiguration(
                        title: "Third Screen",
                        backgroundColor: Color.red,
                        imageName: "eraser.line.dashed.fill"
                    )
                )
            }
            Button("Reset All Screens") { // New
                navPath = []
            }
        }
        .frame(maxWidth: .infinity, maxHeight: .infinity)
        .background(configuration.backgroundColor.opacity(0.2))
        .navigationBarTitle(configuration.title)
    }
}
```

NavigationStack (pros)

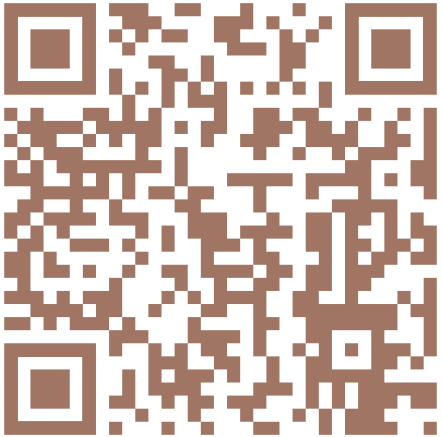
effectiveband

- **Прямой доступ к стеку ViewController**
  - Есть обратная навигация (pop)
  - Есть reset-to-root навигация
- **Есть программная навигация**
  - Можно узнать состояние навигации
  - Можно выставить свое состояние для UI тестов

What’s next?

effectiveband

Apple - Migrating to new navigation types



GitHub - Navigation Backport

GitHub - OzonTech SUINavigation

What’s next?

effectiveband

Medium - NavigationStack vs NavigationView.
(Key Features)

Majid Jabrayilov - Mastering NavigationStack in SwiftUI.
(Deep Linking)

Jenny V - Использование NavigationStack для идеального поведения TabView

Студенческая
ИТ-лаборатория

effectiveband

NavigationStack - Must Have 🚀
NavigationView - Good to Know ✍🏻

Navigation Requirements

effectiveband

- Deeplink Support (app://)
- URL Support (App Scheme)
- Back Navigation
- Gesture Navigation
- Navigation State
- Hidden Realization
- [pop, popTo, skip, isRoot, ...]
- UI Tests Support & Coverage

Meet the Navigation Router

Meet the Navigation Router

effectiveband

Basic Router Example

```swift
import SwiftUI

final class Router: ObservableObject {

    public enum Destination: Codable, Hashable {
        case livingroom
        case bedroom(owner: String)
    }

    @Published var navPath = NavigationPath()

    func navigate(to destination: Destination) {
        navPath.append(destination)
    }

    func navigateBack() {
        navPath.removeLast()
    }

    func navigateToRoot() {
        navPath.removeLast(navPath.count)
    }
}
```

Basic Router Example

```swift
import SwiftUI

@main
struct RoutingApp: App {
    @ObservedObject var router = Router()

    var body: some Scene {
        WindowGroup {
            NavigationStack(path: $router.navPath) {
                HomeView()
                .navigationDestination(for: Router.Destination.self) { destination in
                    switch destination {
                    case .livingroom:
                        LivingroomView()
                    case .bedroom(let owner):
                        BedroomView(ownerName: owner)
                    }
                }
            }
            .environmentObject(router)
        }
    }
}
```

Meet the Navigation Router

effectiveband

Basic Router Example

```swift
import SwiftUI

struct HomeView: View {

    @EnvironmentObject var router: Router

    var body: some View {
        VStack {
            Image(systemName: "house.fill")
                .font(.system(size: 56))
                .foregroundColor(.accentColor)
            Text("Home")
                .font(.system(size: 24))

            Button("Go to Living room") {
                router.navigate(to: .livingroom)
            } label: {
                Text("Go to Livingroom")
            }
            .padding(.top, 12)

        }
        .padding()
    }
}
```

Basic Router Example

```swift
struct BedroomView: View {

    @EnvironmentObject var router: Router
    var ownerName: String

    var body: some View {
        VStack {
            Text("\(ownerName)'s Bedroom")
                .font(.system(size: 36, weight: .bold))
                .padding(.bottom, 12)
            Image(systemName: "bed.double.fill")
                .font(.system(size: 56))
                .foregroundColor(.accentColor)

            Button {
                router.navigateBack()
            } label: {
                Text("Back to Livingroom")
            }
            .padding(.top, 12)

            Button {
                router.navigateToRoot()
            } label: {
                Text("Pop to Home")
            }
            .padding(.top, 4)

        }
        .navigationBarBackButtonHidden()
        .padding()
    }
}
```

Basic Router Example

```swift
import SwiftUI

struct LivingroomView: View {

    @EnvironmentObject var router: Router

    var body: some View {
        VStack {
            Image(systemName: "sofa.fill")
                .font(.system(size: 56))
                .foregroundColor(.accentColor)
            Text("Livingroom")
                .font(.system(size: 24))
                .padding(.top, 12)

            Button("Go to Jane's Bedroom") {
                router.navigate(to: .bedroom(owner: "Jane"))
            }
            .padding(.top, 12)

            Button("Go to John's Bedroom") {
                router.navigate(to: .bedroom(owner: "John"))
            }
            .padding(.top, 12)

            Button {
                router.navigateBack()
            } label: {
                Text("Back")
            }
            .padding(.top, 4)
        }
        .navigationBarBackButtonHidden()
        .padding()
    }
}
```

Какие минусы?

Какие минусы?

effectiveband

- Поддержка и масштабируемость
  - Экраны описаны в Router
  - Можно разбить на несколько
- Нарушение SRP
  - View содержит бизнес логику
  - Как открыть экран в зависимости от состояния?
- Нарушение OCP
  - Всегда приходится указывать .navigationDestination
  - Проблемы с регрессионным тестированием
- Сложное тестирование
  - Не можем протестировать навигацию отдельно от View

Meet the Navigation Router

effectiveband

Basic Router Example

```swift
import SwiftUI

final class Router: ObservableObject {

    public enum Destination: Codable, Hashable {
        case livingroom
        case bedroom(owner: String)
    }

    @Published var navPath = NavigationPath()

    func navigate(to destination: Destination) {
        navPath.append(destination)
    }

    func navigateBack() {
        navPath.removeLast()
    }

    func navigateToRoot() {
        navPath.removeLast(navPath.count)
    }
}
```

Basic Router Example

```swift
import SwiftUI

@main
struct RoutingApp: App {
    @ObservedObject var router = Router()

    var body: some Scene {
        WindowGroup {
            NavigationStack(path: $router.navPath) {
                HomeView()
                .navigationDestination(for: Router.Destination.self) { destination in
                    switch destination {
                    case .livingroom:
                        LivingroomView()
                    case .bedroom(let owner):
                        BedroomView(ownerName: owner)
                    }
                }
            }
            .environmentObject(router)
        }
    }
}
```

Meet the Navigation Router

effectiveband

Basic Router Example

```swift
import SwiftUI

struct HomeView: View {

    @EnvironmentObject var router: Router

    var body: some View {
        VStack {
            Image(systemName: "house.fill")
                .font(.system(size: 56))
                .foregroundColor(.accentColor)
            Text("Home")
                .font(.system(size: 24))

            Button("Go to Living room") {
                router.navigate(to: .livingroom)
            } label: {
                Text("Go to Livingroom")
            }
            .padding(.top, 12)

        }
        .padding()
    }
}
```

Basic Router Example

```swift
struct BedroomView: View {

    @EnvironmentObject var router: Router
    var ownerName: String

    var body: some View {
        VStack {
            Text("\(ownerName)'s Bedroom")
                .font(.system(size: 36, weight: .bold))
                .padding(.bottom, 12)
            Image(systemName: "bed.double.fill")
                .font(.system(size: 56))
                .foregroundColor(.accentColor)

            Button {
                router.navigateBack()
            } label: {
                Text("Back to Livingroom")
            }
            .padding(.top, 12)

            Button {
                router.navigateToRoot()
            } label: {
                Text("Pop to Home")
            }
            .padding(.top, 4)

        }
        .navigationBarBackButtonHidden()
        .padding()
    }
}
```

Basic Router Example

```swift
import SwiftUI

struct LivingroomView: View {

    @EnvironmentObject var router: Router

    var body: some View {
        VStack {
            Image(systemName: "sofa.fill")
                .font(.system(size: 56))
                .foregroundColor(.accentColor)
            Text("Livingroom")
                .font(.system(size: 24))
                .padding(.top, 12)

            Button("Go to Jane's Bedroom") {
                router.navigate(to: .bedroom(owner: "Jane"))
            }
            .padding(.top, 12)

            Button("Go to John's Bedroom") {
                router.navigate(to: .bedroom(owner: "John"))
            }
            .padding(.top, 12)

            Button {
                router.navigateBack()
            } label: {
                Text("Back")
            }
            .padding(.top, 4)
        }
        .navigationBarBackButtonHidden()
        .padding()
    }
}
```

Как масштабировать?

Data Models

effectiveband

Advanced Router Example

```
import Foundation

struct User {
    let uid: UUID
    let name: String
    let donatedValue: Double
    let balance: Double
}
```

Advanced Router Example

```
import Foundation

class Datasource {
    static var mockUser = User(
      uid: UUID(),
      name: "John",
      donatedValue: 46.998,
      balance: 68.477
      )
}
```

Протоколы (интерфейсы)

effectiveband

Advanced Router Example

```
import SwiftUI

protocol ViewFactory {
    func makeView() -> AnyView
}
```

Advanced Router Example

```
protocol NavigationCoordinator {
    func push(_ path: any Routable)
    func popLast()
    func popToRoot()
}
```

Advanced Router Example

```
typealias Routable = ViewFactory & Hashable
```

Точка входа в приложение (после/до)

effectiveband

Advanced Router Example

```swift
import SwiftUI

@main
struct RoutingExampleApp: App {
    var body: some Scene {
        WindowGroup {
            CustomNavigationView(appRouter: .init())
        }
    }
}
```

Basic Router Example

```swift
@main
struct RoutingApp: App {
    @ObservedObject var router = Router()

    var body: some Scene {
        WindowGroup {
            NavigationStack(path: $router.navPath) {
                HomeView()
                .navigationDestination(for: Router.Destination.self) { destination in
                    switch destination {
                    case .livingroom:
                        LivingroomView()
                    case .bedroom(let owner):
                        BedroomView(ownerName: owner)
                    }
                }
            }
            .environmentObject(router)
        }
    }
}
```

Главные сущности

Advanced Router Example

```swift
import SwiftUI

struct CustomNavigationView: View {

    @StateObject var appRouter: AppRouter

    var body: some View {
        NavigationStack(path: $appRouter.paths) {
            appRouter.resolveInitialRouter().makeView()
                .navigationDestination(for: AnyRoutable.self) { router in
                    router.makeView()
                }
        }
    }
}
```

Advanced Router Example

```swift
import SwiftUI

class AppRouter: ObservableObject {
    @Published var paths: NavigationPath

    init(paths: NavigationPath = NavigationPath()) {
        self.paths = paths
    }
}

extension AppRouter: NavigationCoordinator {
    func push(_ router: any Routable) {
        DispatchQueue.main.async {
            let wrappedRouter = AnyRoutable(router)
            self.paths.append(wrappedRouter)
        }
    }

    func popLast() {
        DispatchQueue.main.async {
            self.paths.removeLast()
        }
    }

    func popToRoot() {
        DispatchQueue.main.async {
            self.paths.removeLast(self.paths.count)
        }
    }

}
```

Главные сущности

Advanced Router Example

```swift
import SwiftUI

struct CustomNavigationView: View {

    @StateObject var appRouter: AppRouter

    var body: some View {
        NavigationStack(path: $appRouter.paths) {
            appRouter.resolveInitialRouter().makeView()
                .navigationDestination(for: AnyRoutable.self) { router in
                    router.makeView()
                }
        }
    }
}
```

Type ‘any Routable’ (aka ‘any ViewFactory & Hashable’) cannot conform to ‘Hashable’.

Advanced Router Example

```swift
import SwiftUI

class AppRouter: ObservableObject {
    @Published var paths: NavigationPath

    init(paths: NavigationPath = NavigationPath()) {
        self.paths = paths
    }
}

extension AppRouter: NavigationCoordinator {
    func push(_ router: any Routable) {
        DispatchQueue.main.async {
            let wrappedRouter = AnyRoutable(router)
            self.paths.append(wrappedRouter)
        }
    }

    func popLast() {
        DispatchQueue.main.async {
            self.paths.removeLast()
        }
    }

    func popToRoot() {
        DispatchQueue.main.async {
            self.paths.removeLast(self.paths.count)
        }
    }

}
```

Protocol self-conformance

effectiveband

Type ‘any Protocol’ cannot conform to ‘Protocol’ error.

Type erasure позволяет обернуть тип протокола в какой-то конкретный, который скрывает исходный, но при этом сохраняет его функциональность.

Этот паттерн необходим, когда вы работаете с протоколами как с типами, одновременно поддерживая возможность реализовывать эти протоколы.

Advanced Router Example

```swift
import SwiftUI

// MARK: - A type-erased wrapper for Routable

struct AnyRoutable: Routable {
    private let base: any Routable
    private let equals: (any Routable) -> Bool

    init<T: Routable>(_ routable: T) {
        base = routable
        equals = { other in
            guard let otherBase = other as? T else { return false }
            return routable == otherBase
        }
    }

    func makeView() -> AnyView {
        self.base.makeView()
    }

    func hash(into hasher: inout Hasher) {
        self.base.hash(into: &hasher)
    }

    static func == (lhs: AnyRoutable, rhs: AnyRoutable) -> Bool {
        lhs.equals(rhs.base)
    }
}
```

Time to navigate some things...

effectiveband

Advanced Router Example

```
import SwiftUI
  class AppRouter: ObservableObject {
    func resolveInitialRouter() -> any Routable {
        let homePageRouter = HomePageRouter(rootCoordinator: self, user: Datasource.mockUser)
        return homePageRouter
    }
  }
```

Реализация роутера (пример)

effectiveband

Advanced Router Example

```
extension HomePageRouter: Routable {

    func makeView() -> AnyView {
        let viewModel = HomePageViewModel(router: self, user: user)
        let view = HomePageView(viewModel: viewModel)
        return AnyView(view)
    }
}

extension HomePageRouter {
    static func == (lhs: HomePageRouter, rhs: HomePageRouter) -> Bool {
        lhs.user.uid == rhs.user.uid
    }

    func hash(into hasher: inout Hasher) {
        hasher.combine(self.user.uid)
    }
}

extension HomePageRouter {
    static let mock: HomePageRouter = .init(
      rootCoordinator: AppRouter(),
      user: Datasource.mockUser
    )
}
```

Advanced Router Example

```
import SwiftUI

class HomePageRouter {
    private let rootCoordinator: NavigationCoordinator
    var user: User

    init(rootCoordinator: NavigationCoordinator, user: User) {
        self.rootCoordinator = rootCoordinator
        self.user = user
    }

    func routeToMakeDonationPage() {
        let router = MakeDonationRouter(
            rootCoordinator: self.rootCoordinator,
            user: self.user
         )
        rootCoordinator.push(router)
    }
}
```

Реализация View (пример)

Advanced Router Example

```swift
import Foundation

class HomePageViewModel: ObservableObject {
    private let router: HomePageRouter
    @Published var user: User

    init(router: HomePageRouter, user: User) {
        self.router = router
        self.user = user
    }

    func navigateToMakeDonationPage() {
        self.router.routeToMakeDonationPage()
    }
}

extension HomePageViewModel {
    static let mock: HomePageViewModel = .init(
      router: HomePageRouter.mock,
      user: HomePageRouter.mock.user
    )
}
```

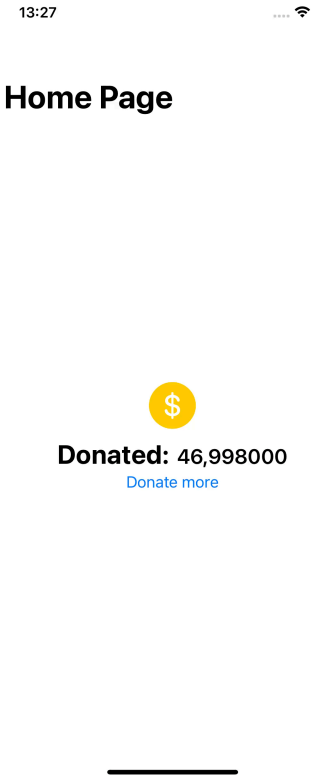
Advanced Router Example

```swift
import SwiftUI

struct HomePageView: View {
    @StateObject var viewModel: HomePageViewModel

    var body: some View {
        VStack {
            Image(systemName: "dollarsign.circle.fill")
                .resizable()
                .aspectRatio(contentMode: .fit)
                .foregroundStyle(.yellow)
                .frame(width: 50)
            HStack(alignment: .firstTextBaseline) {
                Text("Donated:")
                    .font(.title)
                    .fontWeight(.bold)
                Text("\(viewModel.user.donatedValue)")
                    .font(.title2)
                    .fontWeight(.semibold)
            }
            Button(action: {
                self.viewModel.navigateToMakeDonationPage()
            }, label: {
                Text("Donate more")
            })
        }
        .navigationTitle("Home Page")
    }
}
```

Try yourself

Medium - Flexible and Scalable routing approach with StackView

GitHub - RoutingExample



Basic Approach & Type-Erasing

Medium - Routing & Navigation in SwiftUI the right way

Medium - Type-Erasing Protocols

Как меня найти (если нужно)

effectiveband

Telegram

@vectormiller

Email

victorglebovv@yandex.ru

Виктор Глебов
iOS Developer at Effective