OZON tech

SwiftUI навигация: просто, нативно и декларативно

Сергей Балалаев, руководитель отдела
разработки мобильного приложения «ПВЗ»

9 ноября 2023

Сергей Балалаев

Руководитель отдела разработки мобильных приложений «ПВЗ»

Карьера:

- **14 лет** опыта в мобильной разработке
- **9 лет** руковожу людьми
- **8 лет** преподаю курсы iOS

Сергей Балалаев

Руководитель отдела разработки
мобильных приложений «ПВЗ»

OZONtech

Задачи в Ozon:

- Координация и планирование
- Формирование стека технологий
- Построение архитектуры
- Постановка технических задач
- iOS комитет
- Route 256
- Написание статей на Habr
- Выступления на митапах

План

Маршрут презентации

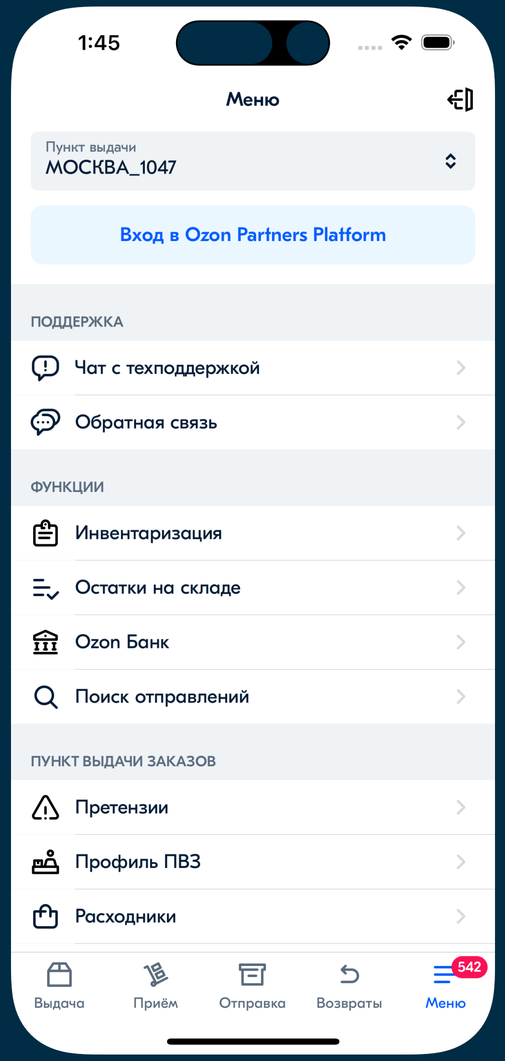
1. iOS SDK
   - iOS 13: NavigationView
   - iOS 16: NavigationStack
   - Бекпорты и альтернативы
2. Решение ПВЗ
   - Декларативность и нативность
   - Ленивая загрузка, управление стеком
   - Баги
3. Заключение
   - DeepLinks

Что за навигация такая?

Navigation

Presentation

OZON tech

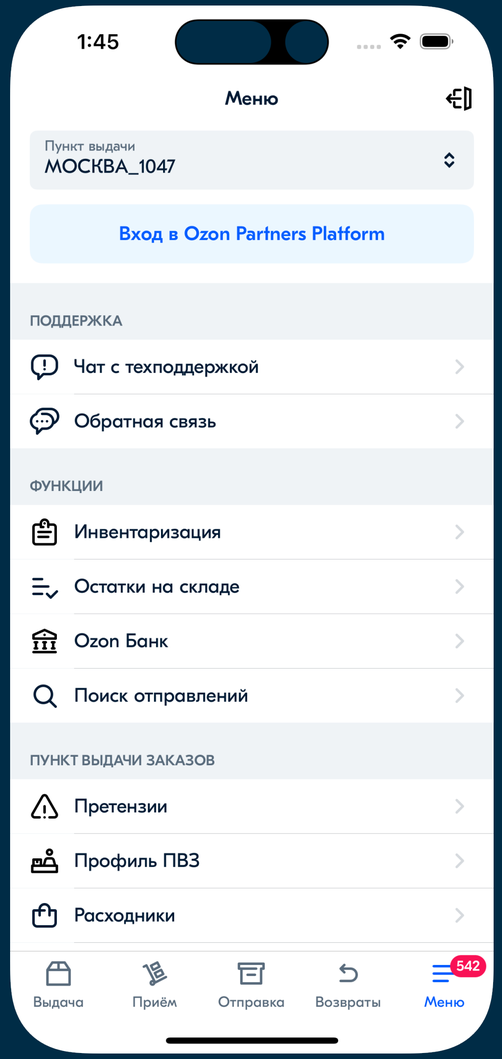


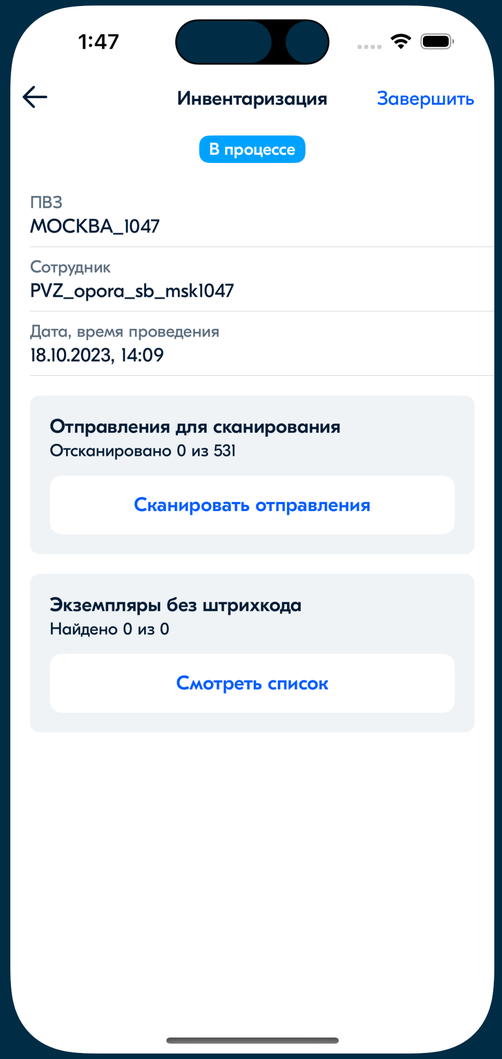
TabBar

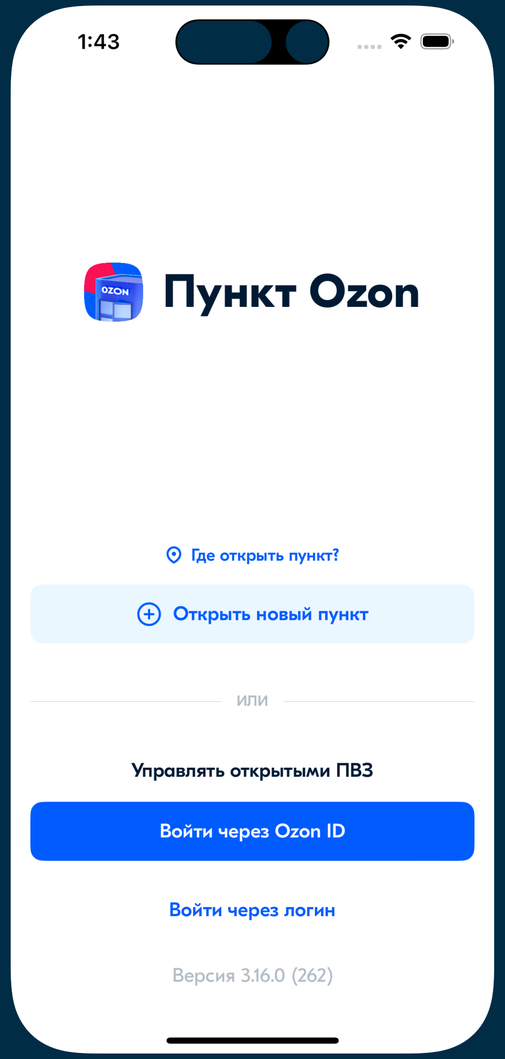
Navigation

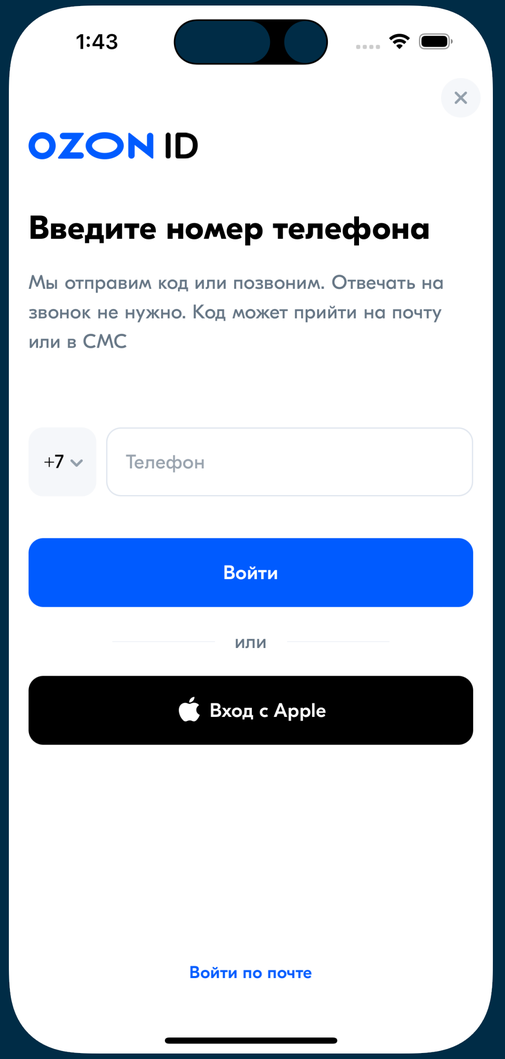
Presentation

OZONtech









NavigationBar, NavigationController

Alert, Sheet, FullScreen

OZONtech

NavigationLink iOS 13

```
struct ContentView: View {
    var body: some View {
        NavigationView {
            MasterView()
        }
    }
}

struct MasterView: View {
    @State private var showDetail = false

    @State var detailsData: DetailsData!

    var body: some View {
        VStack {
            NavigationLink(
                   destination: DetailsView(),
                   isActive: $showDetail)
              {
                Text("Push")
            }
        }
    }
}
```

.fullScreenCover iOS 14

```
struct DetailsData : Identifiable {
    var id: String
}


struct MasterView: View {
    @State private var detailsData: DetailsData? = nil

    var body: some View {
        VStack {
            Text("Nothing")
        }.fullScreenCover(item: $detailsData) { data in
            DetailsView(data: data)
        }
    }
}

struct DetailsView: View {
    @State var data: DetailsData

    var body: some View {
        Text("data")
    }
}
```

1

iOS SDK

Навигация SwiftUI

OZON tech

iOS 13-15

NavigationView

NavigationLink

iOS 16-17

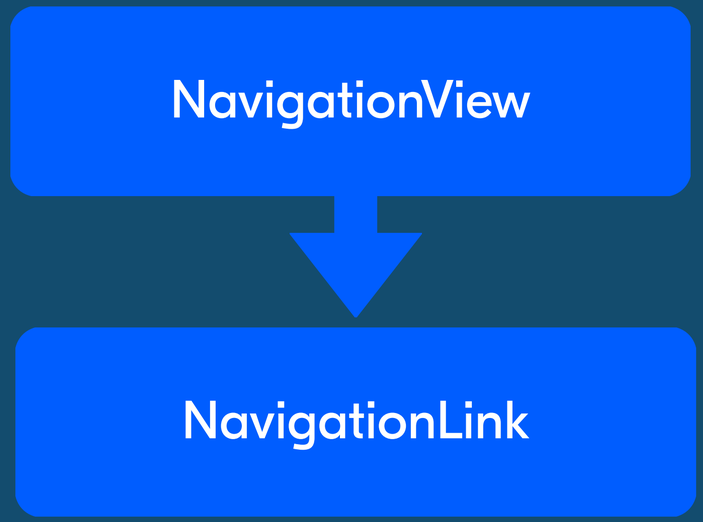
NavigationStack

.navigationDestination

Навигация SwiftUI

OZON tech

iOS 13-15




```swift
import SwiftUI

struct ContentView: View {
    var body: some View {
        NavigationView {
            MasterView()
        }
    }
}

struct MasterView: View {
    @State private var showDetail = false

    var body: some View {
        VStack {
            NavigationLink(destination: DetailsView(),
                           isActive: $showDetail)
            {
                Text("Push")
            }
        }
    }
}
```

Навигация SwiftUI

iOS 13-15

NavigationView

NavigationLink

```
        }
    }
}

struct MasterView: View {
    @State private var showDetail = false

    var body: some View {
        VStack {
            NavigationLink(destination: DetailsView(),
                           isActive: $showDetail)
            {
                Text("Push")
            }
        }
    }
}
```

```
struct DetailsView: View {
    @Binding var showSelf: Bool

    var body: some View {
        Button(action: {
            self.showSelf = false
        }) {
            Text("Pop")
        }
    }
}
```

Навигация SwiftUI

iOS 13-15

NavigationView

NavigationLink

```
struct DetailsView: View {

    @Environment(\.presentationMode)
    var presentationMode: Binding<PresentationMode>

    var body: some View {
        Button(
            "Here is Detail View. Tap to go back.",
            action: {
                self.presentationMode
                    .wrappedValue.dismiss()
            }
        )
    }
}
```

```
struct DetailsView: View {
    @Binding var showSelf: Bool

    var body: some View {
        Button(action: {
            self.showSelf = false
        }) {
            Text("Pop")
        }
    }
}
```

Недостатки NavigationView

1. Сложная передача данных
   - Через @Binding или @Environment
2. В памяти хранится все дерево
   - Все экраны, на которые еще не перешли
3. Нет замены один на другой экран
   - Есть 3 способа сделать костыли
4. Нет удаления из стека конкретных экранов
   - Можно снимать только верхушку экранов и то, если знать как

Навигация SwiftUI

OZON tech

iOS 13-15

NavigationView

NavigationLink

iOS 16-17

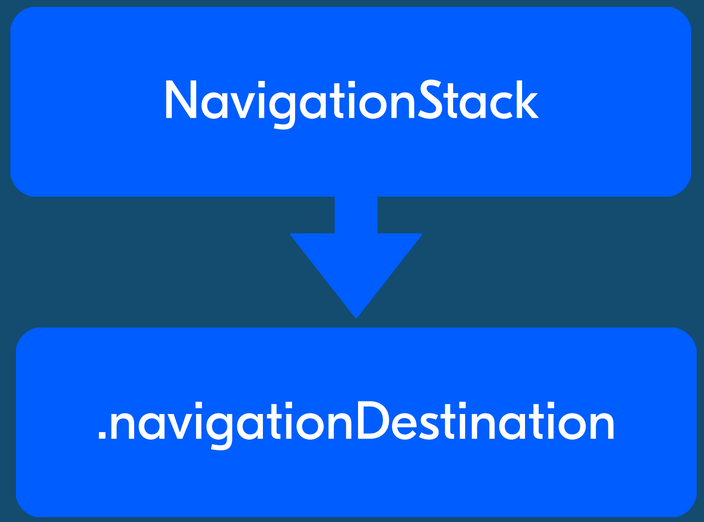
NavigationStack

.navigationDestination

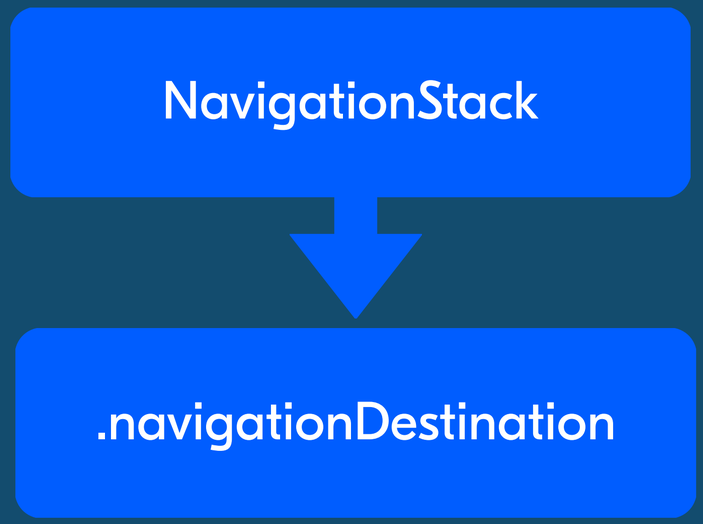
Навигация SwiftUI

OZON tech

iOS 16-17

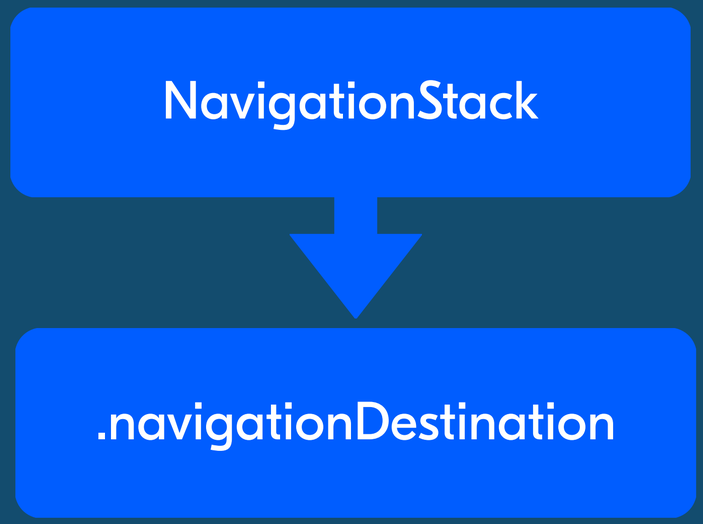


Навигация SwiftUI




```
struct ContentView: View {
  @StateObject private var model = Coordinator()
  var body: some View {

      NavigationStack(path: $model.path) {
          VStack {
              ButtonContent("First View") { model.path = [.firs
              ButtonContent("Second View") { model.path = [.sec
              ButtonContent(“Root View") { model.path = [] }
          }
          .navigationDestination(for: PathItem.self) { destinat
              model.resolve(pathItem: destination)
          }
      }
  }
}

class Coordinator: ObservableObject {
    @Published var path: [PathItem] = []

    func resolve(pathItem: PathItem) -> some View {
        Group {
            switch pathItem {
            case .first: FirstView()
            case .second(let number): SecondView(number: numbe
            }
        }
    }
}
```

Навигация SwiftUI




```
class Coordinator: ObservableObject {
    @Published var path: [PathItem] = []

    func resolve(pathItem: PathItem) -> some View {
        Group {
            switch pathItem {
            case .first: FirstView()
            case .second(let number): SecondView(number: numbe
            }
        }
    }
}
```

```
enum PathItem : Hashable, Identifiable {
  var id: Int {
      return hashValue
  }

  case first
  case second(number: Int)
  case third(string: String)
  case master

}
```

Навигация SwiftUI

iOS 16

NavigationStack

.navigationDestination

```
struct ContentView: View {
  @State private var isShowing = false
  var body: some View {

      NavigationStack() {
          VStack {
              ButtonContent("First View") { isShowing = true }
          }
          .navigationDestination(isPresented: isShowing) {
              FirstView()
          }
      }
   }
}
```

```
struct ContentView: View {
  @StateObject private var model = Coordinator()
  var body: some View {

      NavigationStack(path: $model.path) {
          VStack {
              ButtonContent("First View") { model.path = [.firs
              ButtonContent("Second View") { model.path = [.sec
              ButtonContent(“Root View") { model.path = [] }
          }
          .navigationDestination(for: PathItem.self) { destina
              model.resolve(pathItem: destination)
          }
      }
```

Навигация SwiftUI

ozon tech

iOS 17

NavigationStack

.navigationDestination

```
struct ContentView: View {

  @State private var number: Int? = nil

  var body: some View {

      NavigationStack() {
          VStack {
              ButtonContent("Second View") { number = 1 }
          }
          .navigationDestination(item: $number){ number in
              SecondView(number: number)
          }
      }
  }
}
```

Недостатки NavigationStack

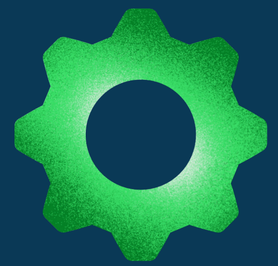
1. Потеря декларативности
   - Координатор и не учитывает строгость графа переходов
2. Поддержка с iOS 16
3. Есть нюансы
   - О багах Apple мы поговорим потом

Есть ли backport и альтернативы?

Backport NavigationStack

OZON tech

1. https://github.com/johnpatrickmorgan/NavigationBackport
2. https://github.com/lm/navigation-stack-backport
3. https://github.com/canopas/UIPilot



NavigationBackport

=

NavigationView
NavigationLink

+

NavigationStack
.navigationDestination

UIKit

Whether interacting with an `Array`, an `NBNavigationPath`, or a `Navigator`, a number of utility functions are available for easier navigation, such as:

```
path.push(Profile(name: "John"))

path.pop()

path.popToRoot()

path.popTo(Profile.self)
```

Note that, if you want to use these methods on an `Array`, ensure the `Array`'s `Element` conforms to `NBScreen`, a protocol that inherits from Hashable without adding any additional requirements. This avoids polluting all arrays with APIs specific to navigation.

Deep-linking

Before `NavigationStack`, SwiftUI did not support pushing more than one screen in a single state update, e.g. when deep-linking to a screen multiple layers deep in a navigation hierarchy. `NavigationBackport` works around this limitation: you can make any such path changes, and the library will, behind the scenes, break down the larger update into a series of smaller updates that SwiftUI supports if necessary, with delays in between. For example, the following code that pushes three screens in one state update will push the screens one by one if needed:

```
path.append(Screen.orders)
path.append(Screen.editOrder(id: id))
path.append(Screen.confirmChanges(orderId: id))
```

Недостатки альтернатив

1. Потеря декларативности
   - Чаще всего заточено на координатор
2. Баги Apple
   - Не все решены
3. UIKit
   - Довольно плотное использование, что тоже ведет к багам в SwiftUI

2

Решение ПВЗ

Пункт Ozon

Приложение для
Пунктов выдачи заказов

Нам 2 года:

- Default компоненты
- Кодогенерация API
- Частично Unit / UI тесты
- SnapshotTests каждого экрана
- Только SwiftUI / Compose
- > 340 экранов и панелек
- Графы переходов очень сложные

Цель

1. Декларативность
   - Направленный граф и автоматическое управление
2. Нативность
   - Не выходим за пределы SwiftUI, приложение как стандартное
3. Ленивая загрузка
   - Догружать дерево экранов в память по мере отображения
4. Возможность управлять стеком
   - Удалять экраны в любой последовательности и количестве
5. Поддержка с iOS 14
   - Ну блин, пишем свою стандартную навигацию кароче

Что такое декларативность?

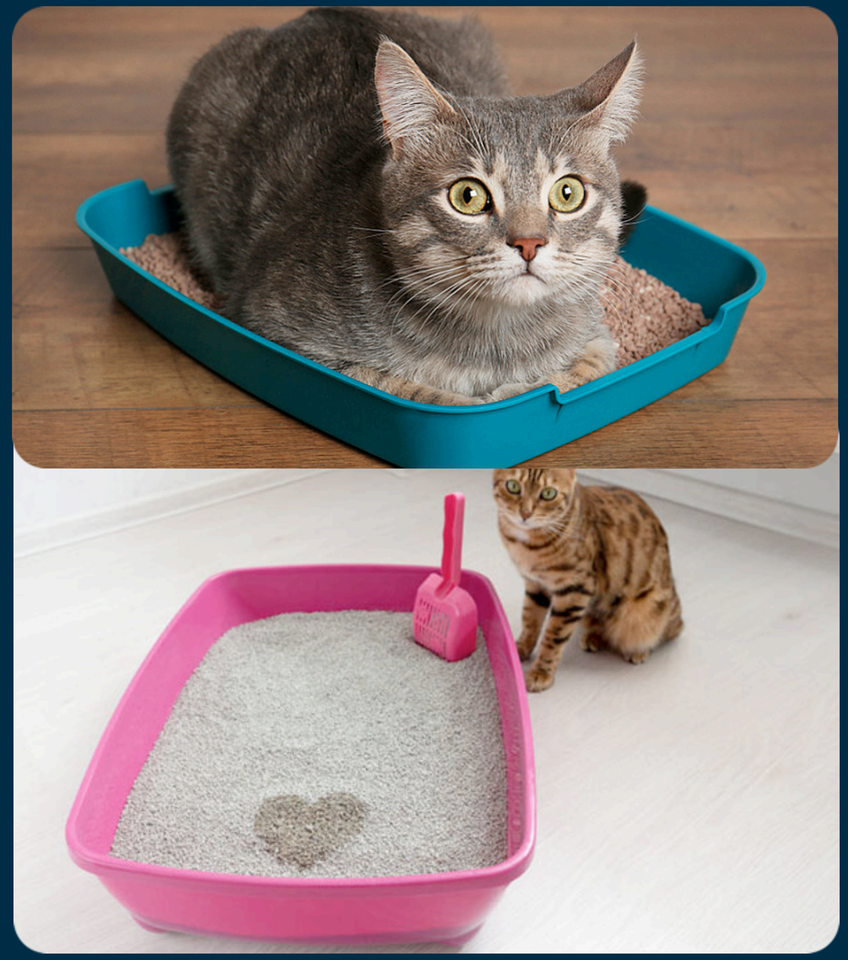
Охота

UIKit

SwiftUI

Санитарные мероприятия

UIKit



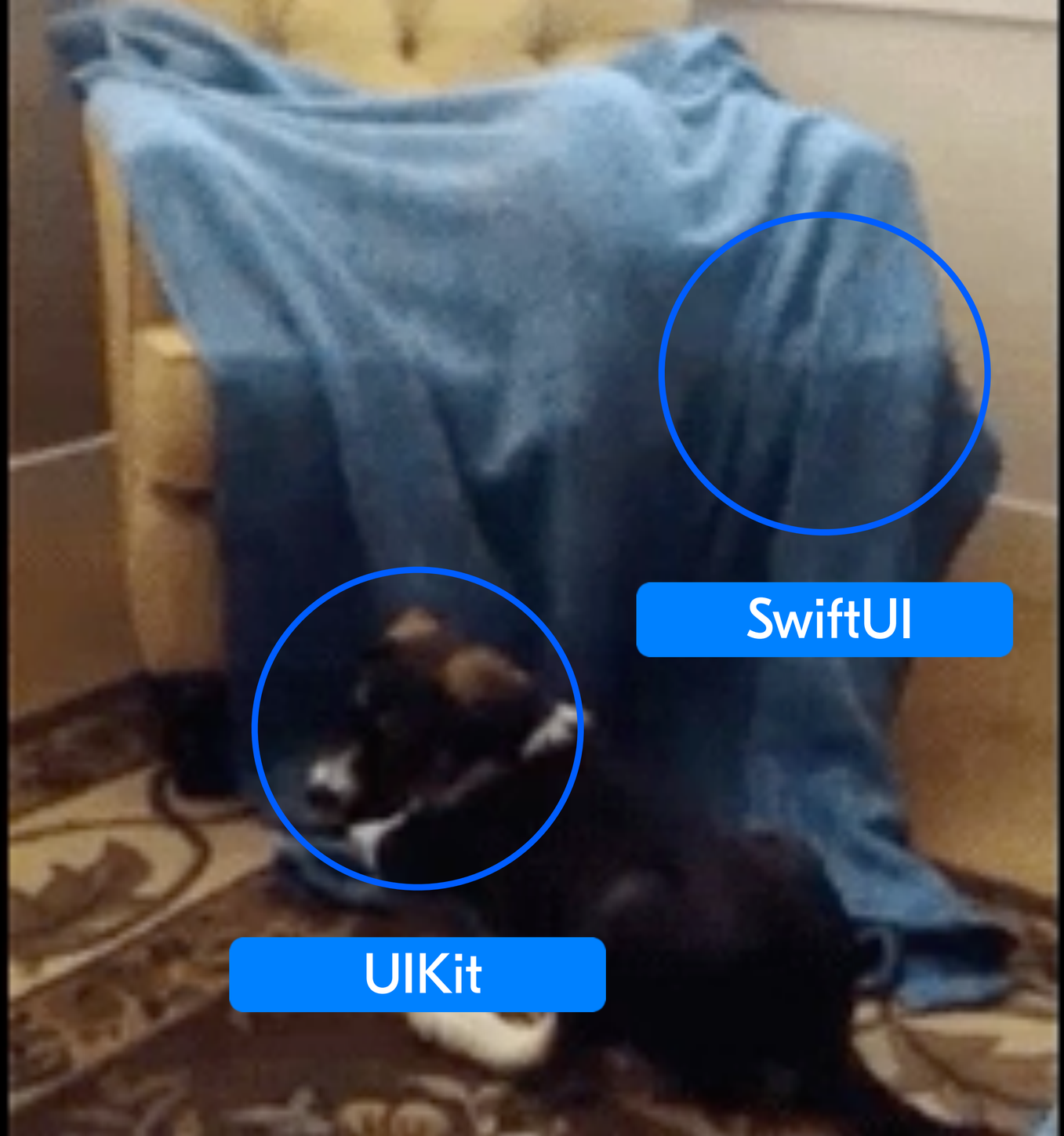
SwiftUI

Гигиена

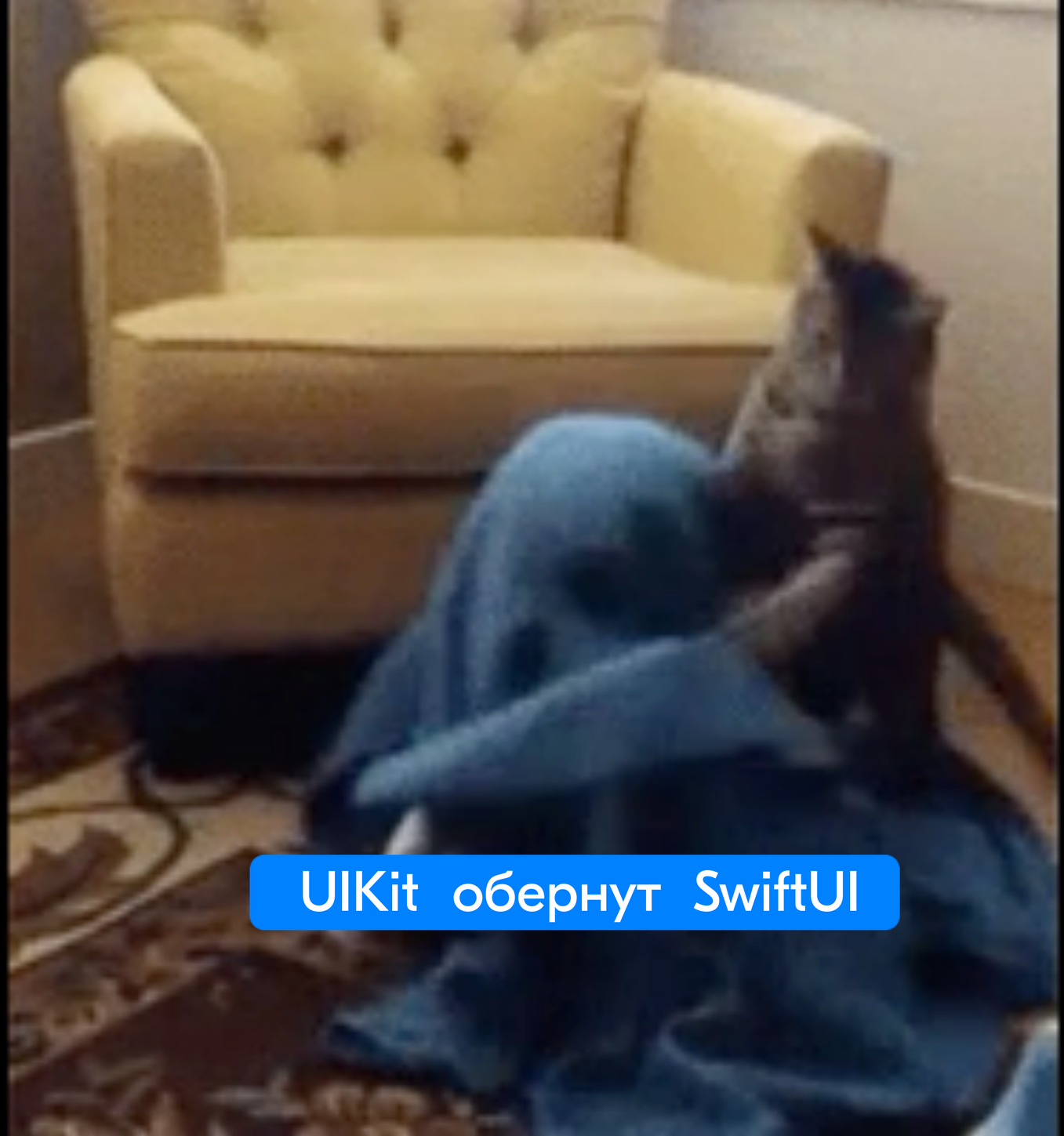
UIKit

SwiftUI

OZON tech

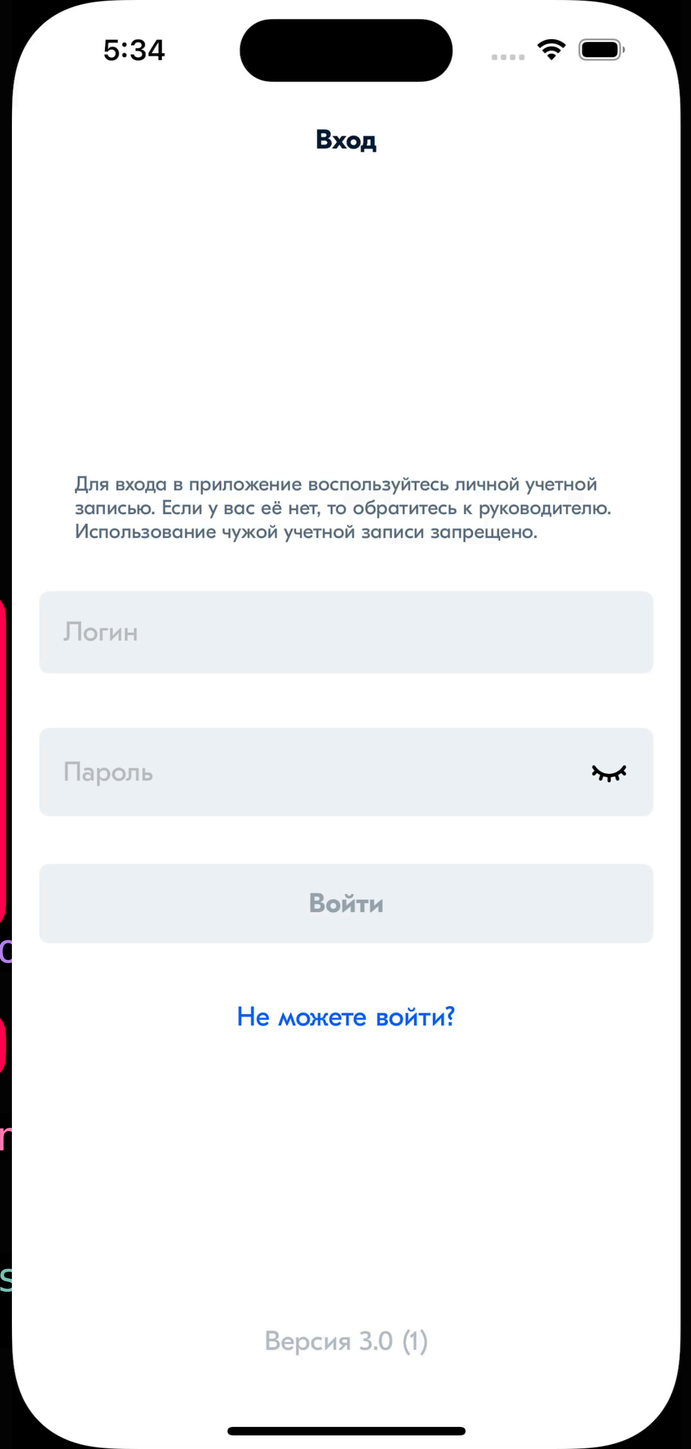


OZONtech



OZONtech

```
var body: some View {
    GeometryReader { geometry in
        if isShowing {
            VStack {
                Spacer()
            }
            .frame(width: geometry.size.width)
            .background(Color(white: 0.3, opacity: 0.6))
            .onTapGesture {
                self.isShowing.toggle()
            }
        }
        VStack(spacing: 0) {
            self.indicator.padding(8)
            self.content().padding(EdgeInsets(top: 0, leading: 0, bottom:
        }
        .frame(width: geometry.size.width, alignment: .top)
        .background(Color.lightGrayColor)
        .cornerRadius(PanelConstants.radius)
        .frame(height: geometry.size.height + bottomOffset, alignment: .bo
        .offset(y:  max( self.offset(geometry) + self.translation, 0))
        .animation(.default, value: translation)
        .gesture(
            DragGesture().updating(self.$translation) { value, state, _ in
                state = value.translation.height
            }.onEnded { value in
                let snapDistance = geometry.size.height * PanelConstants.s
                guard abs(value.translation.height) > snapDistance else {
                self.isShowing = value.translation.height < 0
            }
```



Причем тут Навигация?

SwiftUI: BottomPanel

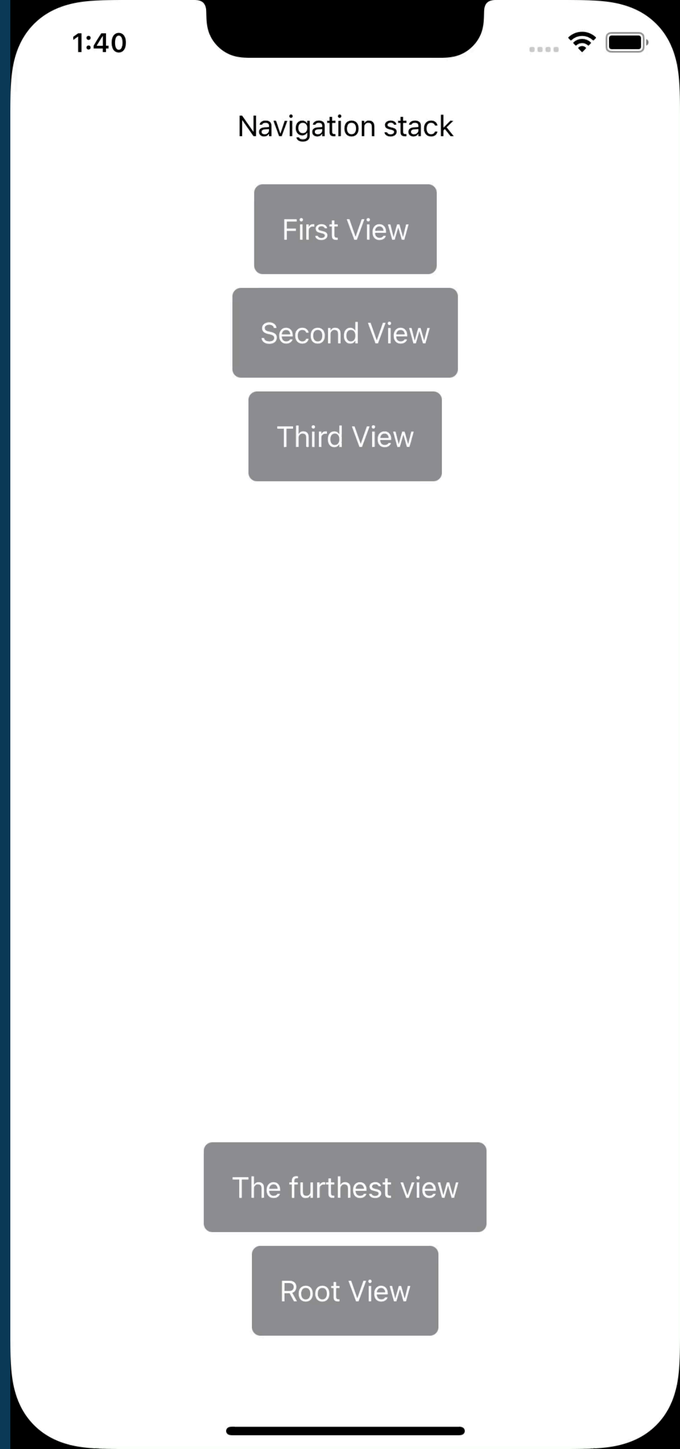
ozon{tech

```
struct BottomPanelView<Content>: View  where Content: View{
  @Binding var isShowing: Bool
  var content: () -> Content

  @GestureState private var translation: CGFloat = 0

  private var bottomOffset : CGFloat {
      PanelConstants.radius * 2.0
  }

  private func offset(_ geometry: GeometryProxy) -> CGFloat {
      isShowing ? 0 : geometry.size.height
  }

  private var indicator: some View {
      RoundedRectangle(cornerRadius: PanelConstants.indicatorHeight / 2.0 )
          .fill(Color.grayColor)
          .frame(
              width: PanelConstants.indicatorWidth,
              height: PanelConstants.indicatorHeight
          ).onTapGesture {
              self.isShowing.toggle()
          }
  }
```

Нативность

.fullScreenCover

```
struct MasterView: View {
  @State private var navigationState: PathItem? = nil

  var body: some View {
      VStack {
          ButtonContent("First View") {
              navigationState = .first
          }
          ButtonContent("Second View") {
              navigationState = .second(number: 8)
          }
      }.fullScreenCover(item: $navigationState) { pathItem in
          switch pathItem {
              case .first: FirstView()
              case .second(let number): SecondView(number: number)
          }
      }
  }
}
```

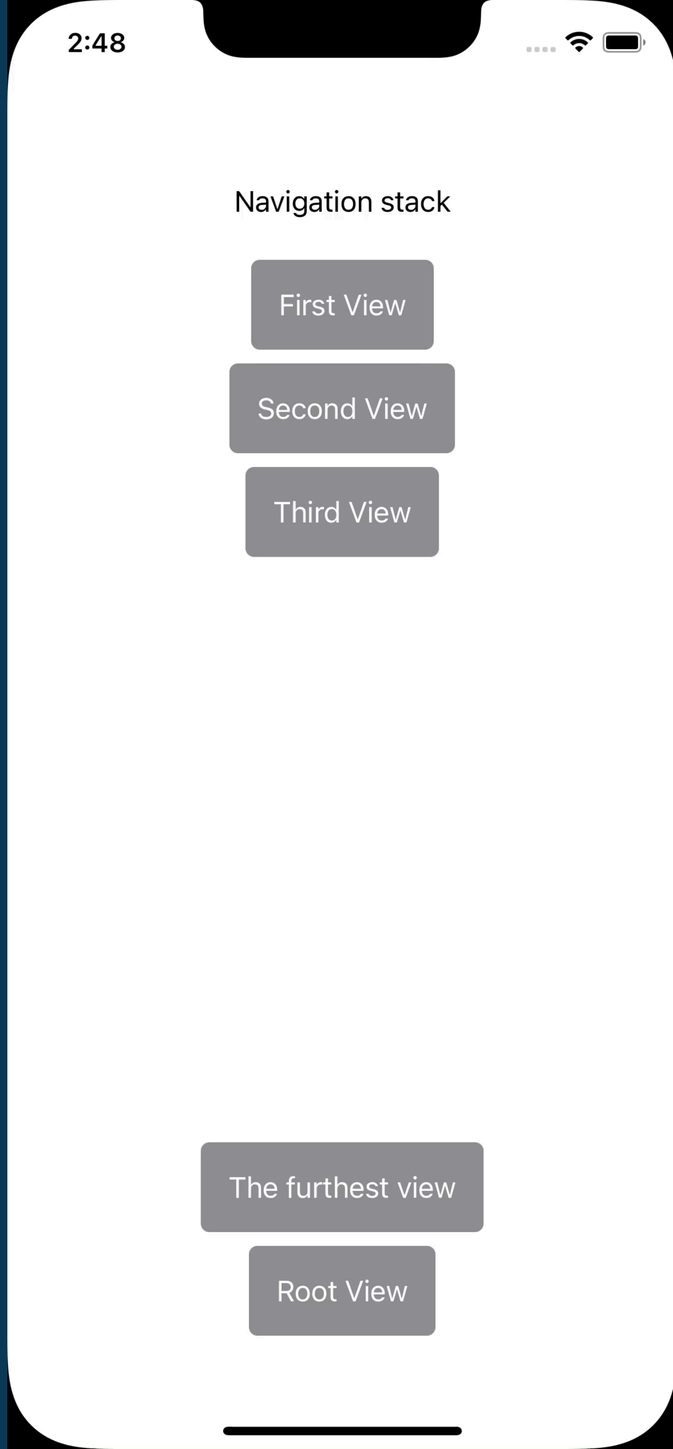


Достоинства .fullScreenCover

1. Нативен
   - По определению
2. Декларативен
   - Учитывает строгость графа переходов и совместим с координатором
3. Ленивая загрузка
   - Clouser вызывается только в момент изменения состояния Identifiable

.fullScreenCover to .navigation

```
struct MasterView: View {
  @State private var navigationState: PathItem? = nil

  var body: some View {
      VStack {
          ButtonContent("First View") {
              navigationState = .first
          }
          ButtonContent("Second View") {
              navigationState = .second(number: 8)
          }
      }.navigation(item: $navigationState) { pathItem in
          switch pathItem {
              case .first: FirstView()
              case .second(let number): SecondView(number: number)
          }
      }
  }
}
```



Ленивая загрузка

```swift
private struct NavigationItemModifier<Destination: View, Item: Equatable>: ViewModifier {
    var item: Binding<Item?>
    @ViewBuilder var destination: (Item) -> Destination
    @State var isActive: Bool = false

    var body: some View {
        NavigationLinkWrapperView(
            id: id,
            destination: navigationDestination,
            isActive: $isActive,
            param: param,
            navigationStorage: navigationStorage
        )
        .onChange(of: item.wrappedValue) { newValue in
            if let newValue {
                isActive = true
            } else {
                isActive = false
            }
        }
        .onChange(of: isActive) { newValue in
            if newValue == false {
                item.wrappedValue = nil
            }
        }
    }
}
```

Как уйти от флагов

```
private struct NavigationItemModifier<Destination: View, Item: Equatable>: ViewModifier {
    var item: Binding<Item?>
    @ViewBuilder var destination: (Item) -> Destination
    @State var isActive: Bool = false

    var body: some View {
        NavigationLinkWrapperView(
            id: id,
            destination: navigationDestination,
            isActive: $isActive,
            param: param,
            navigationStorage: navigationStorage
        )
        .onChange(of: item.wrappedValue) { newValue in
            if let newValue {
                isActive = true
            } else {
                isActive = false
            }
        }
        .onChange(of: isActive) { newValue in
            if newValue == false {
                item.wrappedValue = nil
            }
        }
    }
}
```

Возможность управлять стеком

NavigationStorage

OZON tech

```
final class NavigationStorage: ObservableObject {
  private final class Item: Identifiable, CustomStringConvertible, Hashable {
      let id: String
      let isPresented: Binding<Bool>
      var isSkipped = false

      init(isPresented: Binding<Bool>, id: String) {
          self.isPresented = isPresented
          self.id = id
      }

      static func == (lhs: Item, rhs: Item) -> Bool {
          lhs.id == rhs.id
      }

      func hash(into hasher: inout Hasher) {
          hasher.combine(id)
      }
  }

  @Published
  private var pathItems: [Item] = []
}
```

Общий возврат через экраны

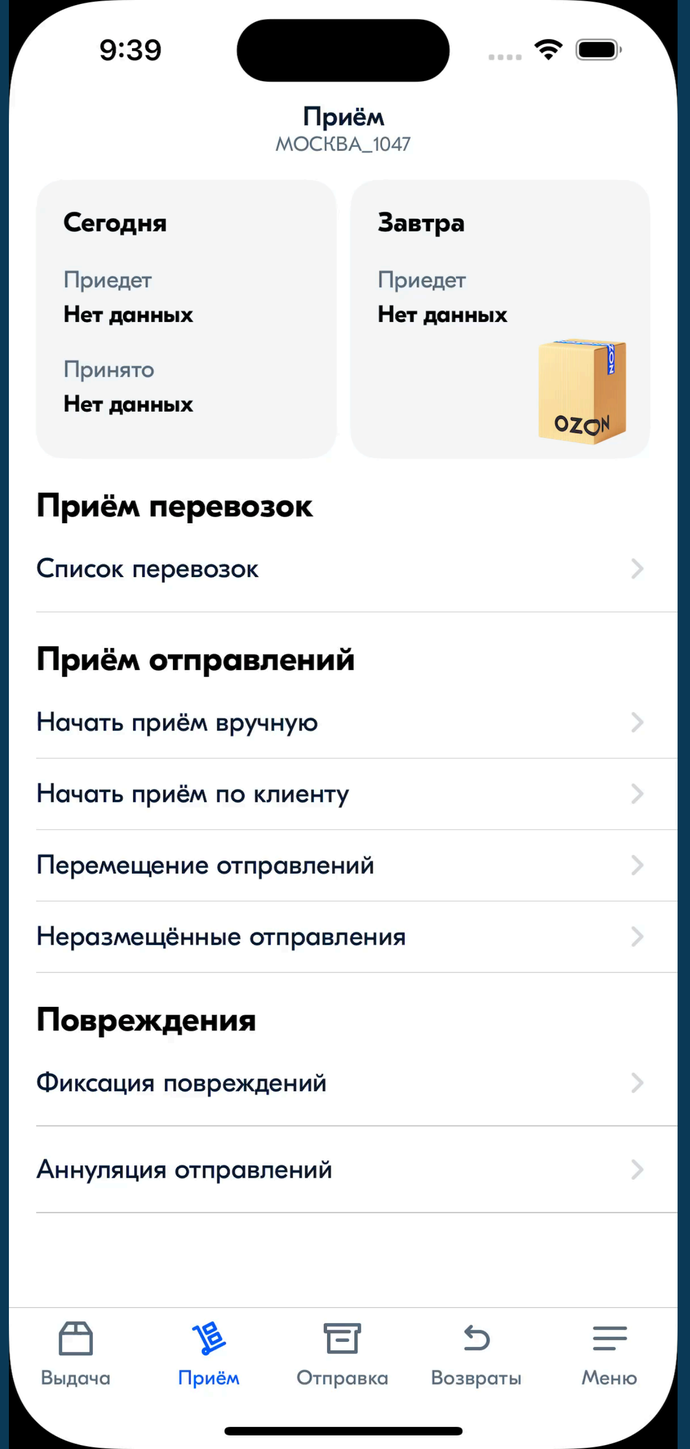
```
private func popTo(_ id: String) {
    guard let foundIndex = pathItems.firstIndex(where: { $0.id == id }) else {
        return
    }
    if #available(iOS 15.0, *) {
        // Work better that original approach but in iOS 14 not stable
        let lastOfItems = pathItems.suffix(from: foundIndex + 1)
        lastOfItems.forEach { pathItem in
            pathItem.isPresented.wrappedValue = false
        }
    } else {
        guard let navController = navigationController(),
            foundIndex + 1 < navController.viewControllers.count else {
            return
        }
        navController.popToViewController(navController.viewControllers[index], animated: true)
    }
    pathItems = Array(pathItems[0...foundIndex])
}
```

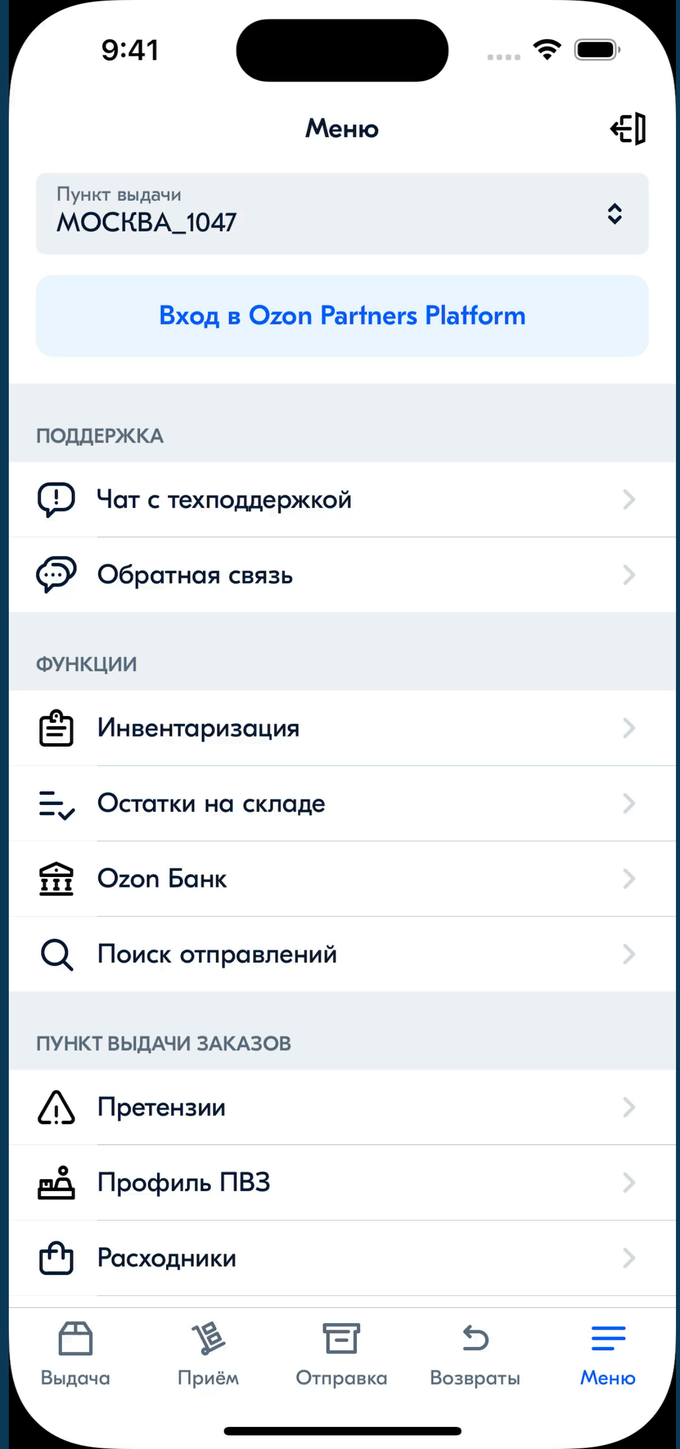
Пропуск экранов при возврате

```
/// In NewView you need call:
/// var body: some View {
///     ...
///     .onApper {
///         navigationStorage.skip(FirstView.self)
///         navigationStorage.skip(SecondView.self)
///     }
/// }
///  and navigation will be automatically
func skip<ViewType: View>(_ type: ViewType.Type) {
    skip(type.navigationID)
}

func skip(_ navigationID: NavigationID) {
    skip(navigationID.rawValue)
}

private func skip(_ id: String) {
    guard let foundItem = pathItems.last(where: { $0.id == id }) else {
        return
    }
    foundItem.isSkipped = true
}
```

Баги NavigationLink

Эффекты от баг Apple





1. NavigationLink: хак от комьюнити

OZON tech

```
@ViewBuilder
var breakView: some View {

    // hacking for supporting iOS 14.5 :
    // https://developer.apple.com/forums/thread/677333

    if #available(iOS 15.0, *) {
        EmptyView().hidden()
    } else {
        NavigationLink(destination: EmptyView()) {
            EmptyView()
        }.hidden()
    }
}
```

2. NavigationLink: и еще один

OZON tech

```
var body: some View {
    breakView
    NavigationLink(
        destination: destination,
        isActive: isActive
    ) {
        EmptyView()
    }
    // bug from Apple: when change screen - dismiss to First View
    // https://developer.apple.com/forums/thread/667460
    .isDetailLink(false)
    .hidden()
    breakView
}
```

3. NavigationView: исправлено только в NavigationStack

ozon{ech

```
public var body: some View {
    NavigationView {
        content
    }
    // bug from Apple: when change screen
    // - dismiss to First View
    // https://developer.apple.com/forums/thread/691242
    .navigationViewStyle(.stack)
    .optionalEnvironmentObject(navigationStorage)
}
```

4. Хак для iOS 14

OZON tech

```
func popTo(_ navigationID: NavigationID) {
    if navigationID == .root {
        popToRoot()
    } else {
        popTo(navigationID.rawValue)
    }
}

func popToRoot() {
    if #available(iOS 15.0, *) {
        pathItems.forEach { pathItem in
            pathItem.isPresented.wrappedValue = false
        }
        pathItems = []
    } else {
        // because we have troubles with iOS 14.5 when stack is big
        navigationController()?.popToRootViewController(animated: true)
        pathItems = []
    }
}
```

5. NavigationView отваливается в iOS 17, но и…

OZON tech

```
@ViewBuilder
private var navigation: some View {
    if #available(iOS 17.0, *) {
        // We can't use it from iOS 16 because
        // The NavigationStack have an issue with dismiss many screens
        // In the stack rest artefact empty screen by this case
        // This issue fixed from iOS 17
        NavigationStack {
            content
        }
    } else {
        NavigationView {
            content
        }
        // bug from Apple: when change screen
        // - dismiss to First View
        // https://developer.apple.com/forums/thread/691242
        .navigationViewStyle(.stack)
    }
}
```

3

Заключение

Заключение

ozon tech

1. SwiftUI без UIKit
   - Нативность, декларативность, ленивость, управляемость, iOS 14 поддерживаемость
2. Поправлены известные баги Apple
   - Нет нужды перекладывать костыли на разработчиков
3. Покрыто UI тестами
   - Что гарантирует стабильность работы
4. Deep Links...
   - Очевидно же что реально!!!

DeepLink

Диплинк — это URI (унифицированный идентификатор ресурсов) или, проще говоря, ссылка, которая отправляет пользователей на конкретную страницу в приложении

Диплинки — это важный способ повышения качества взаимодействия с пользователями за счет переключения пользователей сразу на контент в приложении.

Цель

1. Сохранять
   - Получать состояние URI из текущего состояния навигации
2. Восстанавливать
   - Открыть весь стек экранов из URI
3. Организация Universal Links
   - Чтоб как в определении: «повысить качество взаимодействия»

Поддержка DeepLinks

```
struct FirstView: View {

  @State
  private var numberForSecond: Int? = nil

  @State
  private var isBoolShowed: Bool = false

  var body: some View {
      VStack {
          Button("to Second with 22") {
              numberForSecond = 22
          }
      }
      .navigationAction(isActive: $isBoolShowed){
          BoolView()
      }
      .navigationAction(item: $numberForSecond) { numberValue in
          SecondView(number: numberValue)
      }
  }
}
```

navigationAction = navigation + navigateUrlParams

```
.navigationAction(item: $numberForSecond) { numberValue in
    SecondView(number: numberValue)
}
```

```
.navigation(item: $numberForSecond) { numberValue in
    SecondView(number: numberValue)
}
.navigateUrlParams("SecondView"){ path in
    if let numberForSecond = path.popIntParam(“SecondView") {
        self.numberForSecond = stringForFirst
    }
}
```

- TabBar
- .alert
- .fullScreenCover
- Custom

Максимальная гибкость

```
.navigationAction(item: $numberForSecond) { numberValue in
    SecondView(number: numberValue)
}
```

RootView/FirstView/SecondView/?RootView=FirstView=text&SecondView=20

```
.navigationAction(item: $numberForSecond, id: "second", paramName: "number") { numberValue in
    SecondView(number: numberValue)
}
```

root/first/second/?rootParam=firstText=text&number=20

Ставим через SPM

SUINavigation

https://github.com/ozontech/SUINavigation

Планы

1. Backports 16 iOS
   - NavigationStack работает быстрее, до сих пор не завезли
2. Отключение анимаций
   - Сейчас любое действие сопровождается анимациями
3. TabBar и Presentation
   - Общие модификаторы для стандартных

OZON tech

Спасибо за внимание

Сергей Балалаев, руководитель
отдела разработки
мобильного приложения «Ozon ПВЗ»