X5 Tech

От натива до SDUI через гибрид

2024

X5 Tech — IT-компания и основной цифровой партнёр торговых сетей и бизнесов X5 Group

X5 Tech является частью X5 Group – ведущей российской продуктовой компании. Компания управляет магазинами «у дома» «Пятёрочка», супермаркетами «Перекрёсток» и жесткими дискаунтерами под брендом «Чижик»

Более 392 000 специалистов разрабатывают решения, которые помогают 372 тысячам сотрудников группы работать с максимальным технологическим комфортом, а миллионам покупателей – быстро и удобно покупать свежие продукты

10 000 Пб объём хранения кластера больших данных

30+ цифровых продуктов и 400+ проектов в работе

324 информационные системы в эксплуатации

1400 физических серверов

X5 Tech

О себе

01.

В мобильной разработке с 2010 года. Пользовался Xcode 3-ей версии

02.

В настоящий момент работаю над развитием и внедрением мобильной виджетной платформы в X5 Tech

О чём доклад

Как пришли к тому, что нужен SDUI в приложении

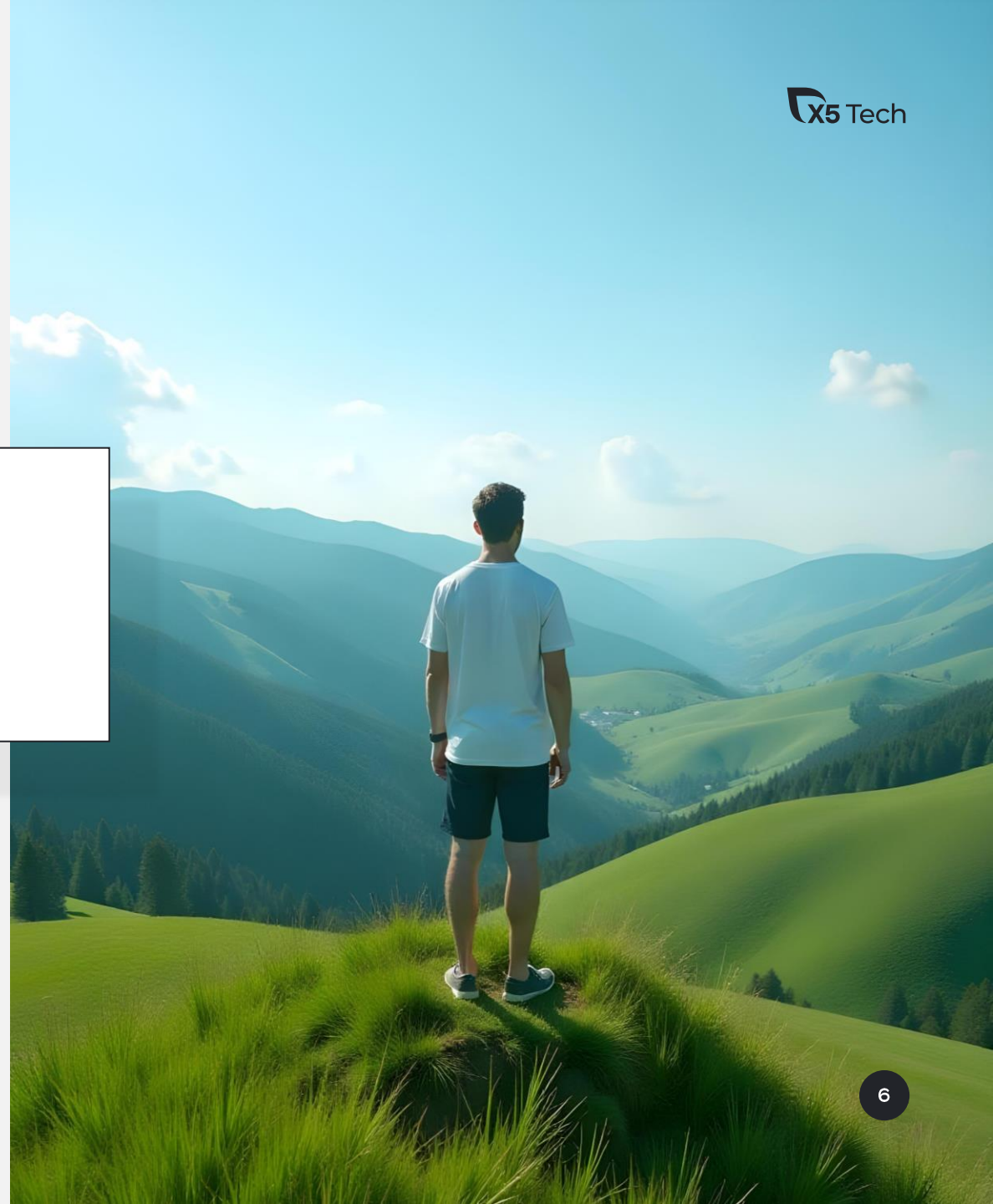
Исходя из чего выбирали решение

Пример взаимодействия виджета и нативной логики

Промежуточный итог внедрения технологии

X5 Tech

В начале перемен



На сцену выходит гибридное приложение

Да

сокращение time to market

Но

пользовательский опыт отличается от нативного приложения

для доступа к нативным функциям требуется разработка собственных решений

service worker доступен только в app bound domain

либо вся страница, либо ничего

А можно UX как в нативе, а TTM как в гибриде?

Да

SDUI

Но

как управлять состоянием

как выполнять конфигурацию интерфейса и пользовательское взаимодействие

где должна находиться бизнес логика приложения

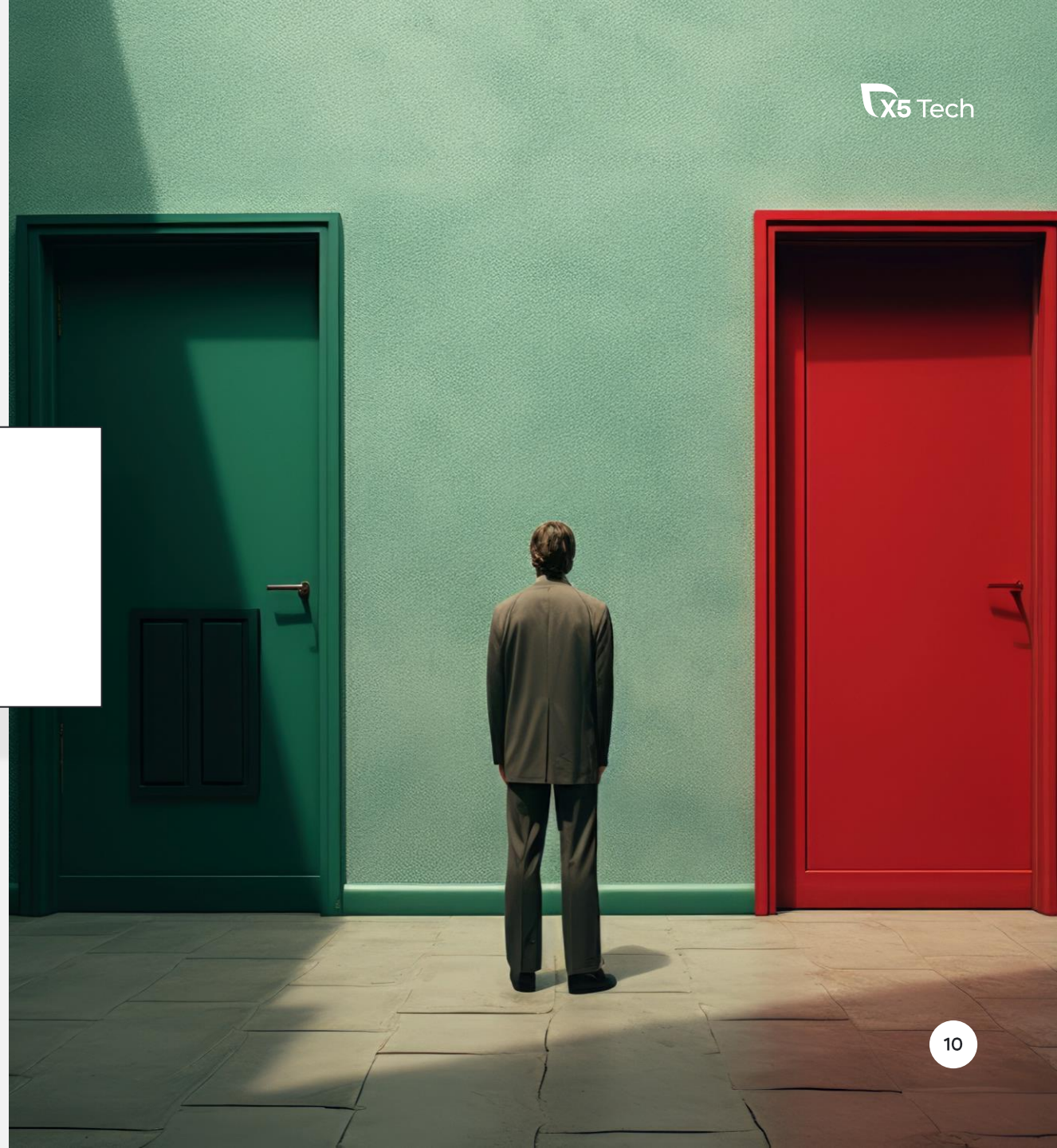
как обеспечить актуальность данных в МП

В идеале

Сделать инструмент, который позволяет реализовать все имеющиеся функции МП, при этом учесть возможность добавления новых фич без релиза в app store

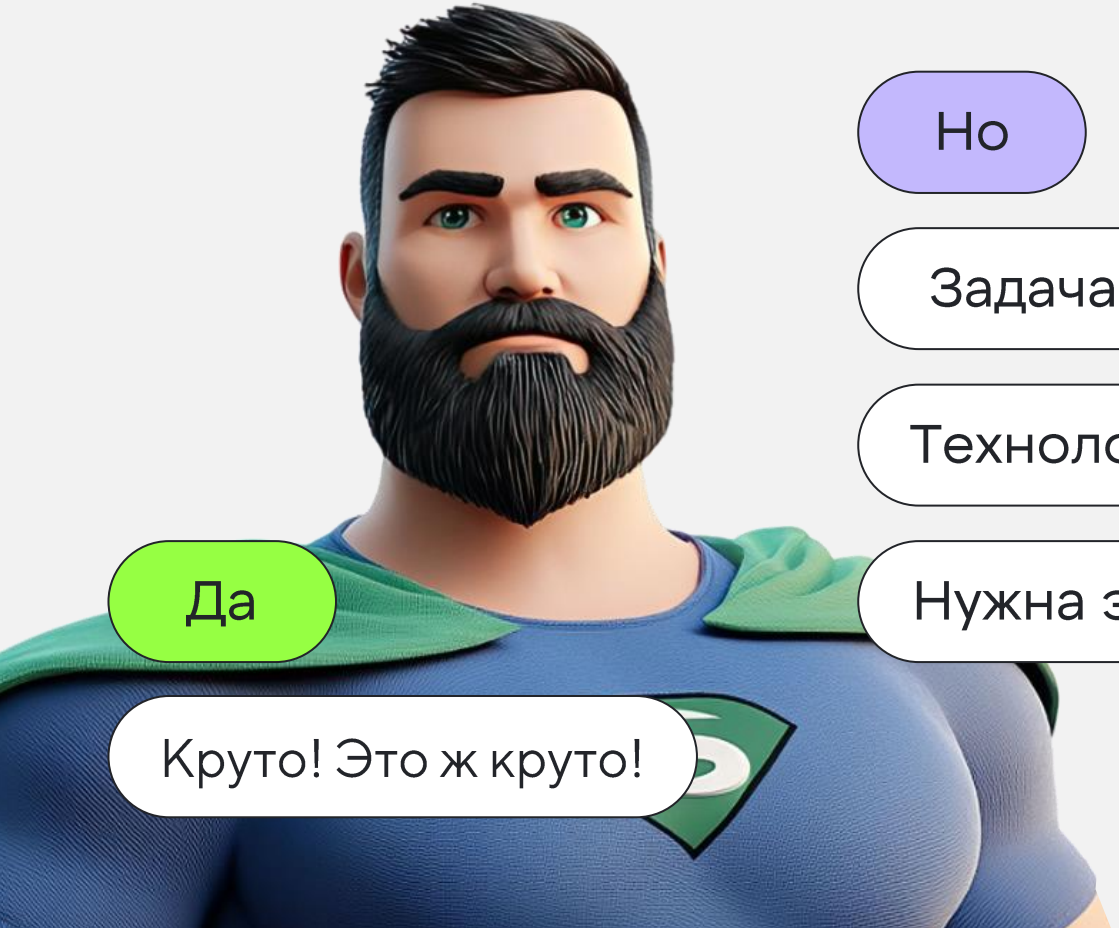
X5 Tech

Давай выберем



2024

Напишем сами – не программисты что ли?!



Да

Круто! Это ж круто!

Но

Задача намного больше, чем на первый взгляд

Технологическая разработка существенно отличается

Нужна экспертиза

Иной релизный цикл

X5 Tech

Ну мы же не первые, кто с этим столкнулся!

Да

Удастся избежать многих ошибок

Быстрый старт

Но

Инструмент может быть как слишком сложным, так и слишком простым

Предвзятое отношение к сторонним решениям внутри команды

Должна быть документация и инструменты для работы с фреймворком

X5 Tech

DivKit

Берешь и делаешь!

Да

текст, изображение, поле ввода, галерея, анимация, таймеры

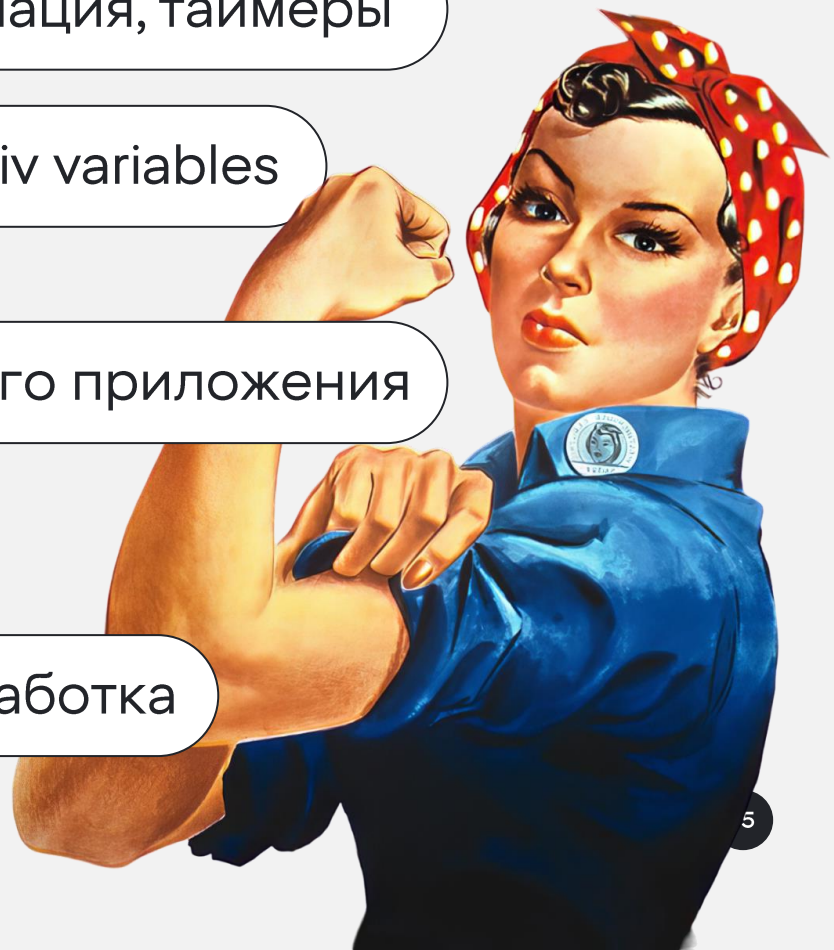
передача состояния между экранами возможна через div variables

Но

все взаимодействие через url-схему мобильного приложения

навигация - через url-схему

что-то специфическое - custom-блок или нативная разработка



X5 Tech

Мы делали апельсин

2024

Всё едет на backend

Да

верстка

навигация между экранами

Но

как быть со сложными экранами и анимацией

что, если требуется оставить нативный элемент

как строить навигацию

дуплексное взаимодействие с нативными модулями

- как актуализировать данные при выполнении пользователем действий
- как передавать состояние между экранами

нет привычной компиляции с подсветкой ошибок, хотя есть валидация JSON’а

URL-схема на все случаи жизни

Да

навигация

передача состояния между экранами

запрос данных с сервера и обновление локального состояния

Но

навигацию нужно унифицировать

юнит-тестами бизнес-логику не покроешь, только служебные классы

поэтому вся надежда на автотесты пользовательских сценариев

Админ-панель, наш философский камень

Да

создание виджетов без написания кода

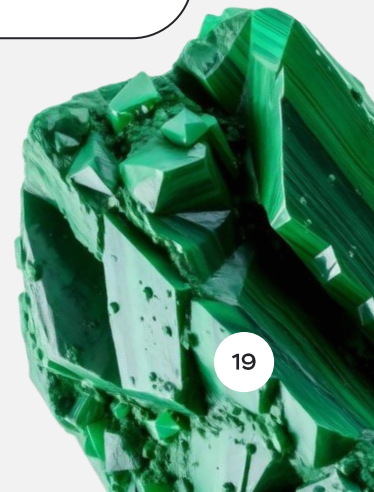
на базе компонентов из дизайн-системы

Но

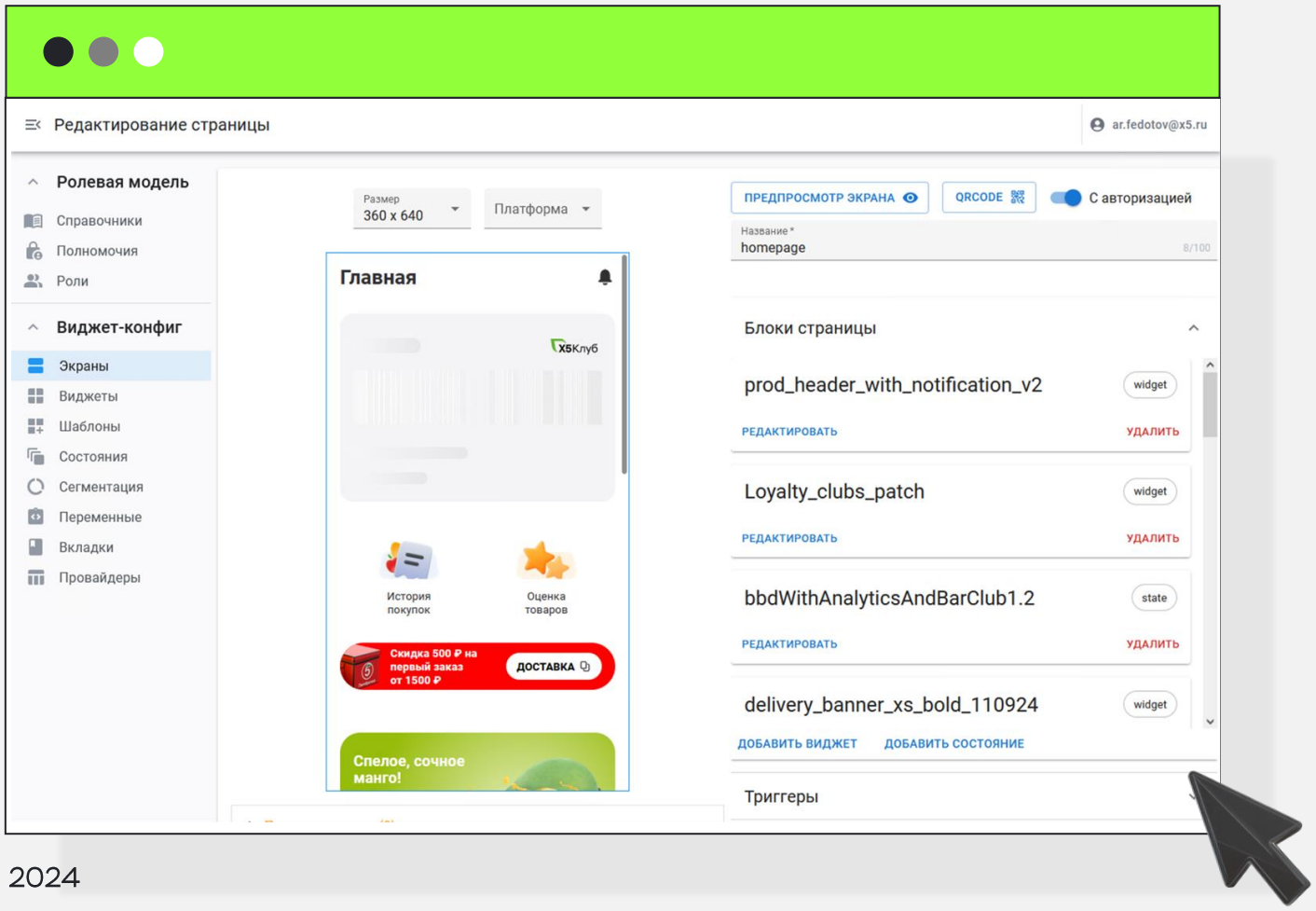
фактически, нужно создать редактор пользовательского интерфейса

который позволяет управлять маппингом данных

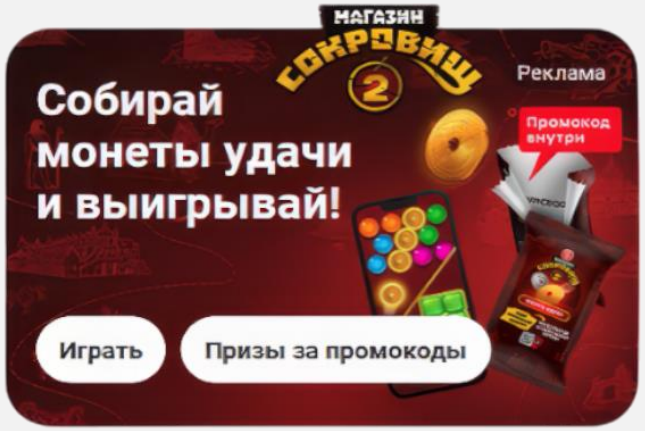
и строить пользовательские сценарии

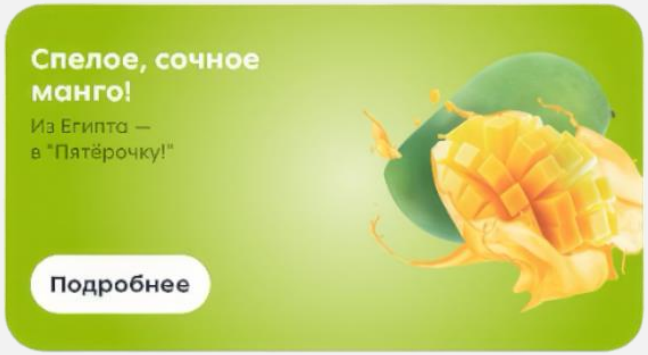


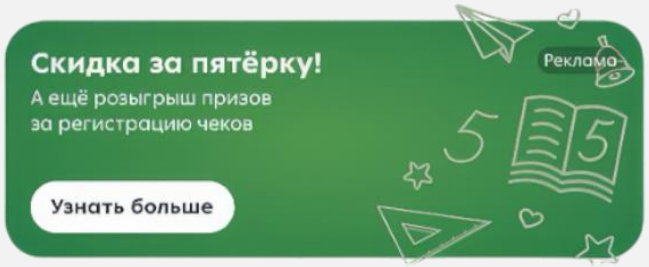
Так выглядит админка для МП



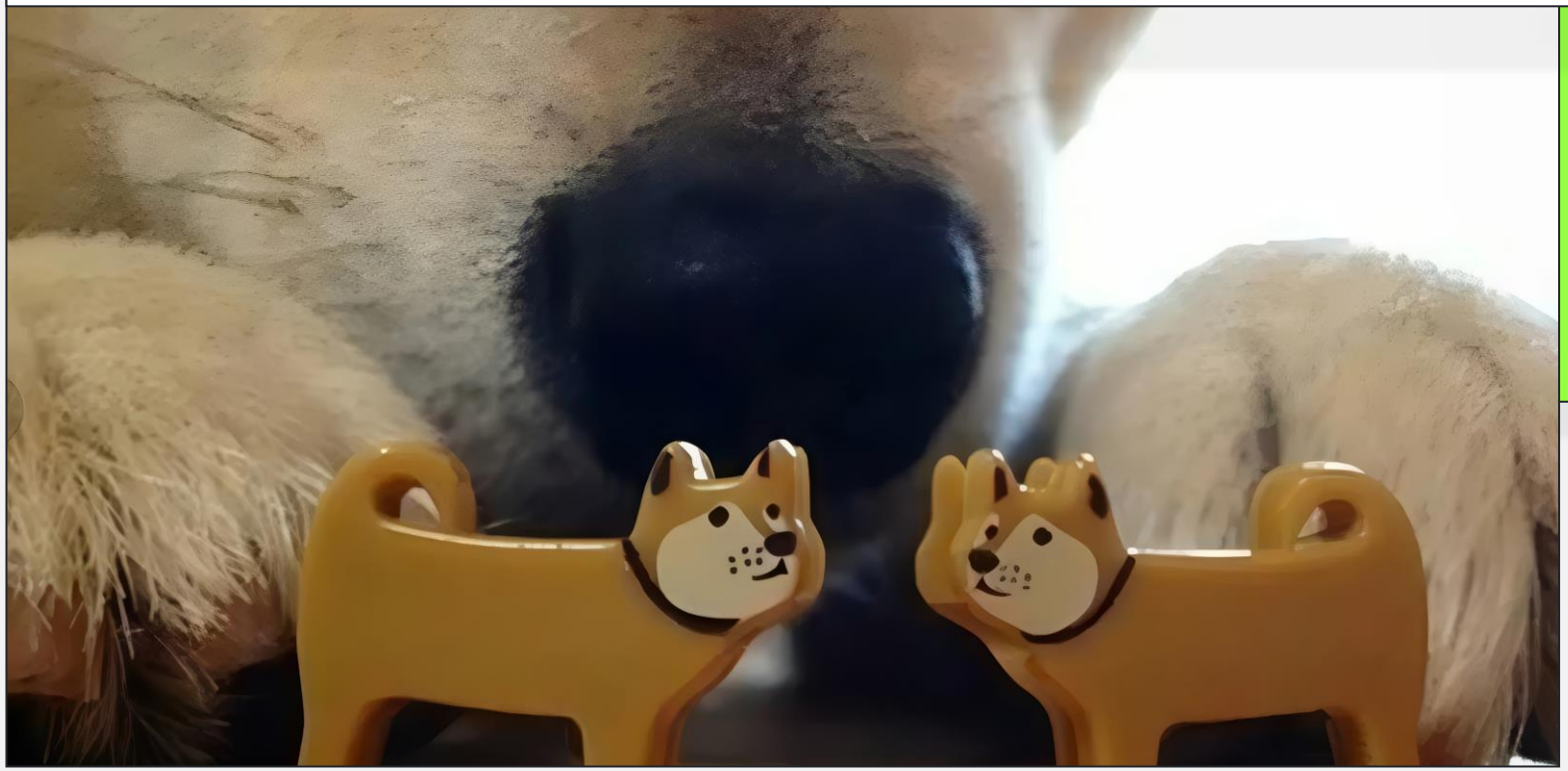
```
{
 "card": {
   "log_id": "sample_config",
   "states": [
     {
       "state_id": 0,
       "div": {
         "type": "container",
         "orientation": "overlap",
         "items": [
           {
             "type": "image",
             "scale": "fill",
             "border": {
               "corner_radius": 24
             },
             "height": {
               "type": "fixed", "value": 216
             },
             "margins": {
               "top": 20, "left": 16, "right": 16, "bottom": 20
             },
             "image_url": "https://s617.../treasureBg.jpg"
           },
           ...
         ],
         ...
       }
     }
   ]
 }
}
```







Пример интеграции виджета с нативной логикой



Что требуется:

01. получение событий из виджета для обработки на стороне приложения

02. отслеживание значения переменной в виджете

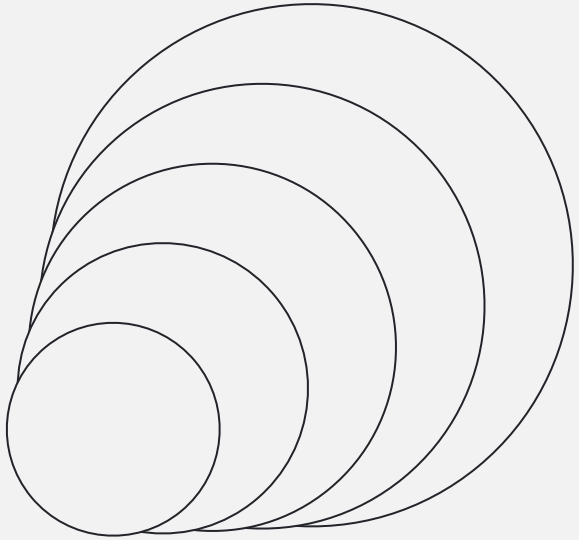
03. изменение состояния виджета

```
{
 ...
 "card": {
   "log_id": "slider_counter_card",
   "variables": [
     {
       "name": "counter",
       "type": "integer",
       "value": 0
     }
   ],
   "states": [
     {
       "state_id": 0,
       "div": {
         ... ,
         "items": [
            {
             "type": "slider",
             "thumb_value_variable": "counter",
             ...
            }
         ]
       }
     }
   ]
 }
}
```

Управление состоянием виджета

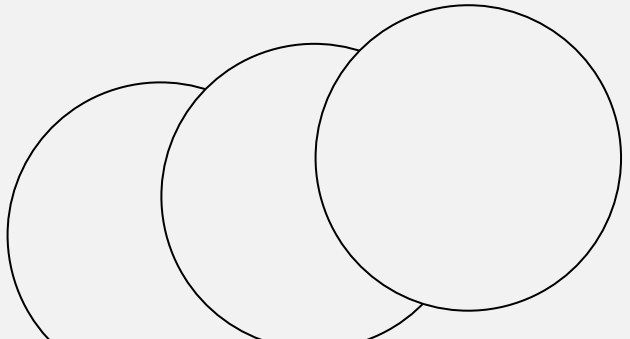
```
final class WidgetProvider {
    private let divKitComponents: DivKitComponents // фасад для взаимодействия с DivKit
    private var divCardID: DivCardID! // уникальный идентификатор виджета
    private var block: Block! // фабрика для создания view

    init(...) { ... }
}
```

```
extension WidgetProvider {
    func setup(_ url: URL, _ view: UIView) async throws {
        // загружаем данные
        let (data, response) = async try URLSession.shared.data(from: url)
        let json: [String: Any] = try JSONSerialization.jsonObject(with: data)
        let divData: DivData = try RawDivData(dictionary: json)

        // инициализируем состояние
        divCardID = divData.logId
        block = divData.makeBlock(context: divKitComponents.makeContext(...))

        // добавляем виджет на экран
        await MainActor.run {
            let blockView = block.makeBlockView()
            view.addSubview(blockView)
        }
    }
}
```



```
extension WidgetProvider {
    func getVariable(name: String) -> String? {
        divKitComponents.variableStorage.getVariableValue(cardId: divCardID, name: name)
    }

    func setVariable(name: String, value: String) {
        divKitComponents.variableStorage.update(cardId: divCardID, name: name, value: value)
    }

    func variablePublisher<T>(name: String) -> AnyPublisher<T?> {
        divKitComponents.variableStorage.addObserver(...)
        // требуется приведение типа к AnyPublisher<T?>, т.к. в DivKit использует собственное Rx-решение
    }
}
```

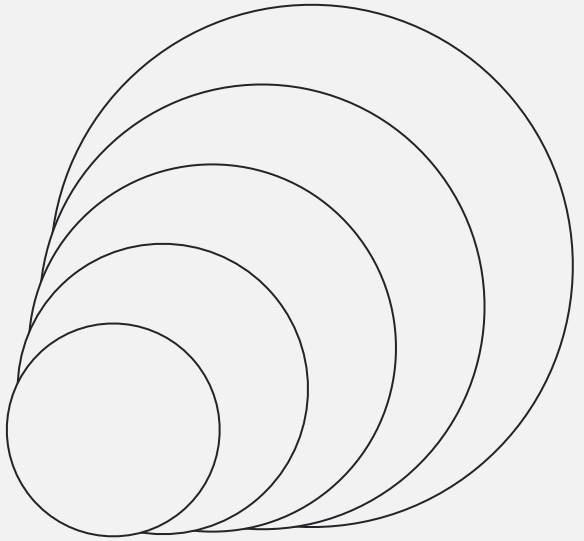
```
final class WidgetViewController: UIViewController {
   var url: URL!
   var widgetProvider: WidgetProvider!

   func viewDidLoad() {
       super.viewDidLoad()

       Task { [weak self] in
           guard let self else { return }

           do {
               try await self.widgetProvider.setup(self.url, self.view)
           } catch {
               debugPrint(error)
           }
       }
   }

   func resetCounterVariable() {
       widgetProvider.setVariable(name: "counter", value: "0")
   }
}
```



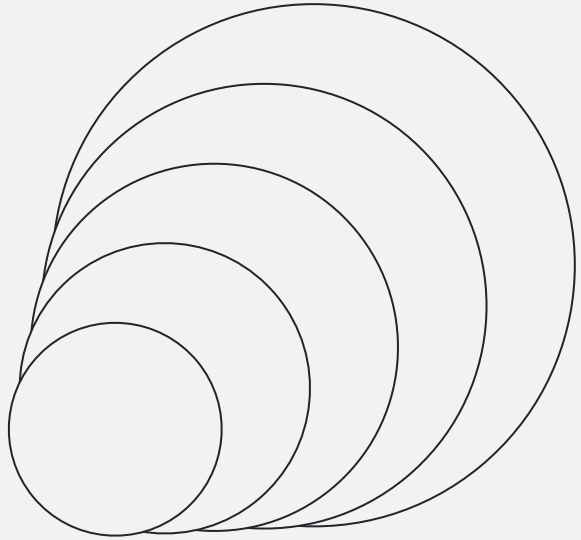
Получение событий из виджета

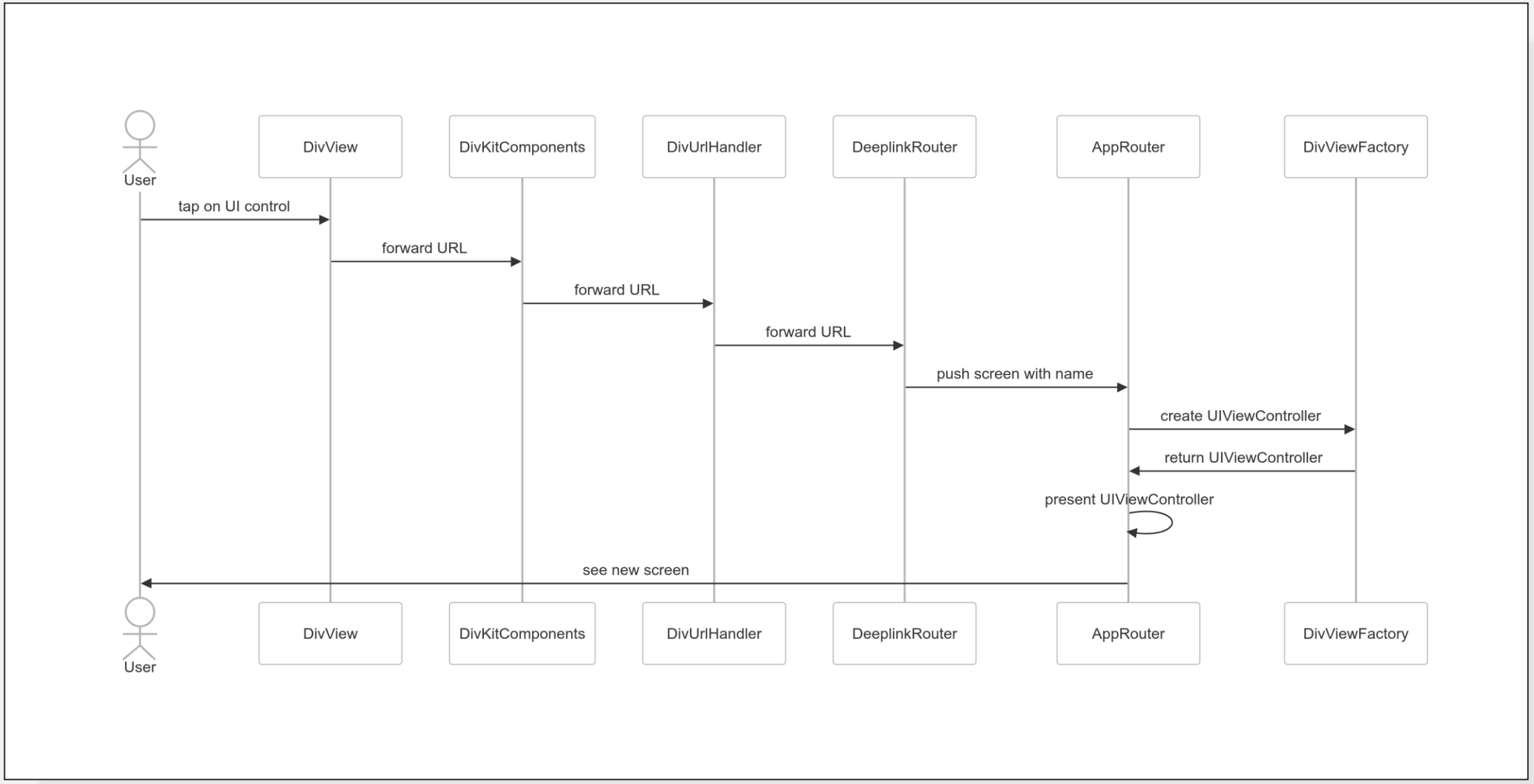
При инициализации DivKitComponents есть возможность передать объект, реализующий протокол DivUrlHandler:

```
public protocol DivUrlHandler {
   func handle(_ url: URL, sender: AnyObject?)
   func handle(_ url: URL, info: DivActionInfo, sender:
AnyObject?)
}

final class DivUrlHandlerImpl: DivUrlHandler {
   let someService: SomeService

   init( ... ) { ... }

   func handle(_ url: URL, sender: AnyObject?) {
       someService.action( ... )
   }
}
```



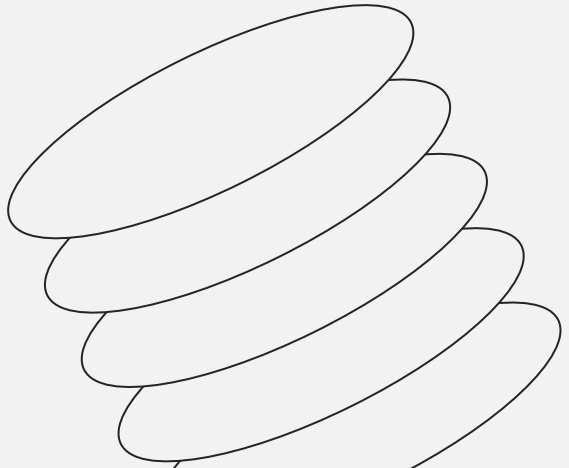
```
{
 "card": {
    ...
    "actions": [
      {
        "url": "app://router/push?screen=@{promoName}",
        "log_id": "button_action"
      }
    ]
 }
}
```

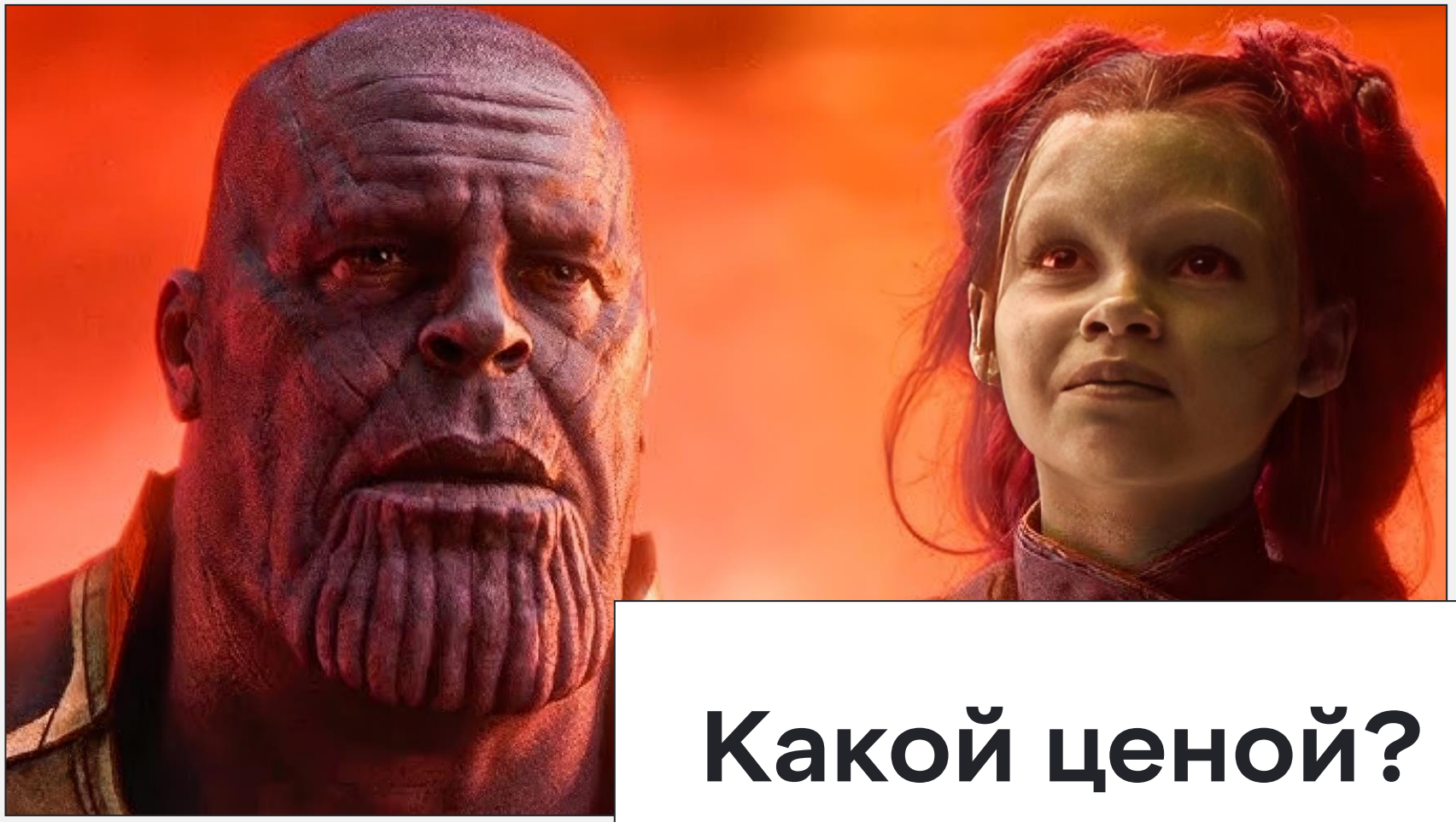



User
DivView
DivKitComponents
DivUrlHandler
DeeplinkRouter
AppRouter
DivViewFactory
tap on UI control
forward URL
forward URL
forward URL
push screen with name
create UIViewController
return UIViewController
present UIViewController
see new screen
User
DivView
DivKitComponents
DivUrlHandler
DeeplinkRouter
AppRouter
DivViewFactory

Инициализация WidgetViewController

```
let widgetProvider = WidgetProvider(
   divKitComponents: DivKitComponents(
       ...
       urlhandler: DivUrlHandlerImpl(
           someService: ...
       )
   )
)

let widgetViewController = WidgetViewController()
widgetViewController.widgetProvider = widgetProvider
```





Какой ценой?

Это сложно!

01. Порог входа достаточно высок

» Отсутствие инструментов приходится компенсировать навыками и эмоциональным ресурсом команды

02. Сильно отличается от привычной мобильной разработки

03. Непонятно, что со специалистами

» kotlin UI developer
» senior JSON developer
» python-верстальщик

Но!

Теперь добавляем новые нативные баннеры сверстанные через админ-панель без релиза мобильного приложения, собираем аналитику и проверяем гипотезы

Вакансии

HR