Mobius

Проблемы геймификации мобильного приложения

Александр Цуцоев

мобильный разработчик KODE

Mobius

Важный дисклеймер

- У нас не было опыта в разработке игр
- Я — мобильный разработчик, а не геймдизайнер
- Аналитик — не геймдизайнер
- UI/UX-дизайнер — не иллюстратор
- Мы использовали свою CI/CD-инфраструктуру — она может отличаться от вашей

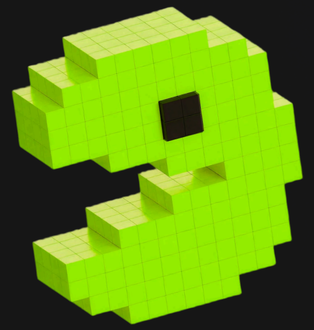
Mobius

От барабана призов до полноценных игр. Или как вообще мы оказались в этой точке

- У бизнеса был успешный опыт внедрения геймификации в виде барабана призов
- Было принято решение развивать идею до игры
- Выбор игры
- UI/UX-дизайнер — не иллюстратор
- У бизнеса было много правок «на ходу»

Mobius

Как мы разрабатывали, с позволения сказать, геймдизайн



Игровой процесс сложнее, чем кажется

Без опыта трудно продумать механику

Технически легко, геймдизайнерски сложно

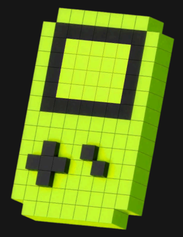
Весёлый и понятный геймплей ≠ пара строк кода

Клон Flappy Bird

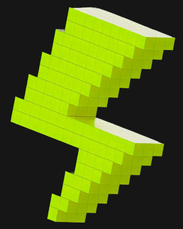
Проблемы начинаются при любых отклонениях

Ограничения, сложность, кастомизация — все требует проработки

QA нужен уже на этапе аналитики



Тестировщики помогают увидеть нестандартные кейсы



Мы не всё учли — пришлось дорабатывать после тестов

Mobius

Выбор технологии

Какие были номинанты

- React native + React Native Reanimated + Skia
- SwiftUI + Compose
- Игровые движки (Unity, React Native Game Engine, Unreal, Godot)

Старый добрый React Native

- опасения по производительности
- много ручной работы
- прекрасно справляется с формами,
  но не с физикой

Нативные реализации

- производительность лучше, чем у RN,
  даже на слабых устройствах
- проблемы остаются: спавн, удаление, коллизии — всё вручную
- никакой кроссплатформенности — код дублируется:
  две игры вместо одной
- любая правка = ×2 времени и усилий

Остаются только игровые движки:

Unreal Unity Godot React Native Game (xe-xe) Engine

Mobius

Выбор технологии → Игровые движки

Unreal

нет экспертизы в C++

Godot

- полностью бесплатный
- легковесный
- gDScript — проще вход

React Native Game Engine

Заброшен

Unity

- популярность и большое комьюнити
- много туториалов
- есть библиотеки для интеграции с React Native
- условно-бесплатная лицензия (внимательно читайте условия)

Mobius

Разработка игры

Освоить Unity быстро и удобно — много документации и туториалов. Важно:

Примеры ассетов от дизайнеров

— нужно адаптировать макеты под текстуры

Обязательно изолированное тестирование

— подготовьте билд под Windows/macOS для команды и бизнеса

Интеграция в приложение

— простой, но достаточно трудоёмкий этап

Mobius

Пример кода на Unity

```
using UnityEngine;

public class Spawner : MonoBehaviour
{
    public GameObject prefab;
    public GameObject prefabWithTopGift;
    public GameObject prefabWithBottomGift;
    public float spawnRate = 2f;
    public float minHeight = -1.5f;
    public float maxHeight = 1.5f;

    private void OnEnable()
    {
     InvokeRepeating(nameof(Spawn), spawnRate, spawnRate);
    }

    private void OnDisable()
    {
        CancelInvoke(nameof(Spawn));
    }

    private void Spawn()
    {
        var pipes = Instantiate(prefab, transform.position, Quaternion.identity);

        pipes.transform.position += Vector3.up * Random.Range(minHeight, maxHeight);
    }
}
```

1 — Что нужно, чтобы выгрузить игру на платформу?

- В Unity откройте File → Build Settings и выберите платформу iOS. Нажмите Switch Platform
- Затем откройте Player Settings. В разделе Other Settings укажите: Bundle Identifier, версию, IL2CPP как Scripting Backend
- Перейдите в раздел Identification и настройте Team ID и Provisioning Profile, если необходимо.
- В проекте добавьте файлы NativeCallProxy.h и NativeCallProxy.mm в Assets/Plugins/iOS.

2 — Что нужно, чтобы выгрузить игру на платформу?

- Вернитесь в Build Settings, нажмите Build, выберите папку (не внутрь RN-проекта!) — Unity сгенерирует Xcode-проект
- Откройте его и выполните ручные настройки:
  — У Data и NativeCallProxy.h измените Target Membership на UnityFramework
  — Установите public для NativeCallProxy.h, чтобы он был доступен
- В Xcode выберите таргет UnityFramework, схему Any iOS Device (arm64) и соберите проект (Cmd + B)
- Найдите полученный UnityFramework.framework в Xcode → Products, щёлкните правой кнопкой, выберите Show in Finder, и скопируйте его в папку unity/builds/ios внутри вашего RN-проекта.

Mobius

Пример кода интеграции

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Runtime.InteropServices;
using UnityEngine.UI;
using UnityEngine;

public class NativeAPI
{
#if UNITY_IOS && !UNITY_EDITOR
 [DllImport("__Internal")]
 public static extern void sendMessageToMobileApp(string message);
#endif
}

public class MessageToReactNative : MonoBehaviour
{
  private const string GameOver = "GameOver";
  private const string LoseLive = "LoseLive";
  private const string Win = "Win";
  private const string Close = "Close";

  private void SendMessageBase(string message)
  {
    if (Application.platform == RuntimePlatform.Android)
    {
      using var jc =
        new AndroidJavaClass("com.azesmwayreactnativeunity.ReactNativeUnityViewManager");
      jc.CallStatic("sendMessageToMobileApp", message);
    }
    else if (Application.platform == RuntimePlatform.IPhonePlayer)
    {
#if UNITY_IOS && !UNITY_EDITOR
      NativeAPI.sendMessageToMobileApp(message);
#endif
    }
  }

  public void SendWin()
  {
    SendMessageBase(Win);
  }
}

}
```

Mobius

Хождение по интеграции

User Interface

- Unity UI — есть, но неудобный и ограниченный
- Потеря темизации приложения при использовании Unity UI
- Рекомендация: создавать UI слоем поверх (overlay) на стороне React Native / нативного приложения

Налаживаем мосты

- Разделение логики — хороший стиль
- Unity принимает конфиг игры из приложения (очки, цели)
- Минимизация общения

Размер файлов

Unity генерирует много промежуточного кода

Android:

~800 МБ

{ много мелких файлов }

iOS

~250 МБ

{ один большой
файл — фреймворк }

Проблема: тяжело хранить в Git

Решение: использовали Git LFS (Large File Storage)

Проблемы с симулятором

- Unity не работает в iOS Simulator
- Пока игра в проекте — симулятор недоступен
- Это серьёзно бьёт по процессу разработки

Mobius

Хождение по интеграции

Трудности начинаются после завершения разработки игры

CI/CD

- Ошибки в CI/CD — одни из самых болезненных
- Подключение unityFramework в iOS
- Установка NDK для Android
- Ошибки CMAKE_C_COMPILER и CMAKE_CXX_COMPILER стали последней каплей

Mobius

Зачем вообще все это, если так и не удалось заставить работать?

Несмотря на финальный отказ от Unity, опыт оказался ценным

Мы вскрыли множество нюансов, о которых почти никто не писал.
Это может помочь тем, кто:

- делает пет-проект
- хочет поэкспериментировать с интеграцией игр
- решится пройти весь путь до продакшена

Хотя игра запускалась из React Native, 95% времени ушло на нативные IDE и настройку платформенной части

Mobius

React Native, прости господи, Game Engine

Плюсы

- простая интеграция
- hot reload
- заработало почти сразу

Минусы

- производительность на грани допустимого
- подозрения на утечки памяти

В поисках идеального решения: Godot

Мы отказались от Godot из-за отсутствия поддержки Android в существующей RN-библиотеке. Сейчас разрабатываю собственную интеграцию

Почему Godot выглядит перспективно

- Проще и легче, чем Unity
- исходники занимают меньше места
- хорошая производительность
- открытый и полностью бесплатный

Что же делать?

Если у вас:

Нативный проект

– простые игры пишите на платформе без привлечения игровых движков

Кросс-платформа

– рассмотрите вариант Godot и своего адаптера с движка на платформы

Что в итоге?

- Несмотря на костыли мы смогли реализовать игру
- Бизнес был доволен и получил увеличение метрик
- Мы провели увлекательный месяц
за разработкой игры

Mobius

Заключение

- Интеграция игры в React Native — реально, но **непросто**
- **React Native + Skia** слабо подходит из-за производительности и отсутствия геймплейной логики, но должно подойти для простых игр
- **Нативные UI-фреймворки** (SwiftUI / Compose) — потребуют дублирования всей логики, но будет хорошо работать
- **Unity** — мощный, но сложный в интеграции, особенно в CI/CD
- **RN Game Engine** — простой и компромиссный, но тормозной и с утечками
- **Godot** — потенциально лучшее решение, но требует доработки

Mobius

Q&A

Александр Цуцоев

мобильный разработчик KODE

@APPKODE

Telegram