Интернет-журнал «Отходы и ресурсы» https://resources.today
Russian Journal of Resources, Conservation and Recycling

2020, №4, Том 7 / 2020, No 4, Vol 7 https://resources.today/issue-4-2020.html

URL статьи: https://resources.today/PDF/13INOR420.pdf

DOI: 10.15862/13INOR420 (http://dx.doi.org/10.15862/13INOR420)

**Ссылка для цитирования этой статьи:**
Недяк А.В., Рудзейт О.Ю., Зайнетдинов А.Р., Рагулин П.Г. Инструменты мобильной кроссплатформенной разработки приложений // Интернет-журнал «Отходы и ресурсы», 2020 №4, https://resources.today/PDF/13INOR420.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ. DOI: 10.15862/13INOR420

**For citation:**
Nedyak A.V., Rudzeyt O.U., Zainetdinov A.R., Ragulin P.G. (2020). Mobile cross-platform app development tools. *Russian Journal of Resources, Conservation and Recycling*, [online] 4(7). Available at: https://resources.today/PDF/13INOR420.pdf (in Russian) DOI: 10.15862/13INOR420

**УДК 004.42**

**ГРНТИ 20.53.01**

**Недяк Антон Витальевич**
ФГОУ ВО «Дальневосточный федеральный университет», Владивосток, Россия
Магистрант
E-mail: nedyaq@gmail.com

**Рудзейт Олег Юрьевич**
ФГОУ ВО «Дальневосточный федеральный университет», Владивосток, Россия
Магистрант
E-mail: rudzeyt18@mail.ru

**Зайнетдинов Артём Ришатович**
ФГОУ ВО «Дальневосточный федеральный университет», Владивосток, Россия
Магистрант
E-mail: zainet.ar@gmail.com

**Рагулин Петр Григорьевич**
ФГОУ ВО «Дальневосточный федеральный университет», Владивосток, Россия
Профессор кафедры «Компьютерных систем»
Кандидат технических наук
E-mail: ragulin_p@mail.ru

# Инструменты мобильной кроссплатформенной разработки приложений

**Аннотация.** В данной статье рассматриваются существующие популярные инструменты для мобильной кроссплатформенной разработки приложений. Она противопоставляется так называемой нативной разработке приложений – когда приложение создаётся при помощи тех инструментов, которые предоставляются компаниями-разработчиками соответствующих платформ. Google для своей мобильной операционной системы Android предоставляет такой инструмент как Android Studio. Основными языками программирования, на котором создаются приложения под эту платформу, являются Java и Kotlin. Apple, в свою очередь, предлагает разработчиками такую интегрированную среду разработки как XCode, в которой разработка ведётся с помощью языка программирования Swift. Авторами были рассмотрены одни из самых популярных инструментов для

кроссплатформенной разработки: React Native, Xamarin и Flutter. React Native – продукт, который разрабатывается компанией Facebook. Он наследует основные технологические аспекты от другого фреймворка, созданного этой же компанией – React. Разработка здесь ведётся на языке программирования JavaScript. React Native подойдёт для веб-разработчиков, которые работали до этого с React и теперь хотят попробовать свои силы в разработке мобильных приложений. Также в данной статье рассмотрен инструмент под названием Xamarin – фреймворк, который на данный момент поддерживается компанией Microsoft. Он предоставляет инструменты для создания кроссплатформенных приложений на мобильные операционные системы iOS и Android. Данный фреймворк поставляется в комплекте с Microsoft Visual Studio в качестве скачиваемого компонента. Что характерно для продукта Microsoft, здесь для разработки используется язык программирования C#. Это является весомым преимуществом фреймворка Xamarin: в ходе создания мобильного приложения разработчики могут использовать все необходимые функции этого языка программирования. В качестве ещё одного инструмента кроссплатформенной разработки авторами был рассмотрен Flutter – это инструмент, разрабатываемый компанией Google для создания приложений для Android и iOS с использованием единой кодовой базы. В отличие от других популярных решений, Flutter не является фреймворком – в данном случае это набор средств для разработки программного обеспечения, который содержит всё необходимое для создания кроссплатформенных приложений. Он включает в себя механизм визуализации, готовые виджеты, а также инструменты для работы с командной строкой. Основной язык программирования здесь – Dart. В ходе изучения данных инструментов, авторами были сделаны следующие выводы: инструменты для кроссплатформенной разработки являются отличным решением в том случае, когда необходимо создать приложение, не требующее высокой производительности: отображение информации, получаемой с сервера, простые локальные обработчики информации, например, приложения для социальных сетей, интернет-магазинов, приложения-органайзеры. Для создания требовательных к вычислительным ресурсам приложений эти инструменты не подходят, в таком случае лучше использовать нативные средства создания мобильных приложений.

**Ключевые слова:** кроссплатформенная разработка; android; iOS; react native; xamarin; flutter; нативная разработка; мобильные операционные системы

## Введение

Разработка мобильный приложений отличается от разработки приложений для ПК (персональных компьютеров). Приложения для мобильных гаджетов должны быть созданы с учётом нюансов, связанных с небольшими размерами устройств (относительно ПК), использованием сенсорного экрана, особенностями дизайна пользовательских интерфейсов (так называемые гайдлайны – требования к дизайну для нативных приложений, описанные разработчиками платформ).

На рынке мобильных операционных систем лидируют два продукта: iOS, разрабатываемая компанией Apple и Android, разрабатываемая компанией Google. Эти мобильные операционные системы относятся к одному семейству – так называемому, семейству Linux. Это означает, что обе системы созданы на ядре Linux, а значит в основе своей являются сходными. Однако создание приложений для этих платформ существенно отличается.

Основное отличие – в языках программирования, на котором пишутся приложения и предоставляемых инструментах разработки. Google для создания приложения под свою операционную систему использует язык программирования Java, а также с мая 2017-го года поддерживается Kotlin. Также Google поддерживает программное обеспечения для работы с

платформой – Android Studio. Чтобы разрабатывать приложения для устройств с операционной системой iOS, компания Apple разработала собственную среду – XCode. Это программное обеспечение, которое предназначено для разработки приложений под все операционные системы компании Apple, в частности для iOS. В роли основного языка программирования здесь используется Swift, ранее также использовался ныне неактуальный язык программирования Objective-C. Важная особенность разработки приложений для iOS: для запуска XCode обязательно требуется устройство с установленной операционной системой MacOS, поскольку не существует версий данной среды для других операционных систем [1].

В большинстве случаев, когда есть задача разработать мобильное приложение, разработка ведётся именно для этих двух платформ: суммарно они занимают 99.37 % рынка мобильных операционных систем по состоянию на ноябрь 2020 года[1]. Поэтому чаще всего, когда компания сама разрабатывает мобильные приложения или предоставляет услуги разработки мобильных приложений для других лиц, то она держит две команды разработчиков – для обеих платформ. Это естественным образом увеличивает издержки – как финансовые, так и временны́е. Также появляется проблема совместной разработки: необходимо минимизировать отличия в приложении для разных операционных систем, при этом учитывая все дизайнерские особенности платформ. Это так называемая разработка нативных приложений, или нативная разработка – для каждой платформы приложение разрабатывается с использованием соответствующих официальных инструментов.

Ей противопоставляется кроссплатформенная разработка – когда приложение создаётся в основном при помощи одного инструмента сразу для двух или более платформ [2]. На рынке представлено большое количество таких инструментов, некоторых из самых популярных которые будут рассмотрены далее.

## React Native

React Native является разработкой компании Facebook, которая построена на базе фреймворка ReactJS – JavaScript-библиотеки для разработки пользовательских интерфейсов. React Native позволяет использовать для разработки мобильных приложений мультипарадигмальный язык программирования JavaScript. Наиболее популярное применение данного языка программирования – написание скриптов для интегрирования в html-код сайтов, обычно для реализации различных интерактивных элементов на веб-страницах [3]. Данным фреймворком поддерживается разработка для мобильных операционных систем Android и iOS.

React Native позволяет использовать нативный код – если необходимо реализовать какой-то объект, создание которого сильно отличается на разных платформах. Приложение строится из нативных модулей, которые «обёрнуты» в React-компоненты [4]. Эти модули бывают как кроссплатформенные (например, для отображения текста и изображений), так и отдельные для Android (например, тулбар – верхняя часть приложения с заголовком приложения и важными кнопками) и iOS (например, навигационная панель в нижней части приложения, которая позволяет переключаться между разными вкладками внутри программы). Фреймворком предоставляются компоненты для реализации прокручиваемых списков, изображений, текста, ввода с клавиатуры.

Важной особенностью является отсутствие компонента для реализации кнопки в приложениях. Однако, есть специальный обработчик «onPress», который может сделать

[1] https://gs.statcounter.com/os-market-share/mobile/worldwide – Mobile Operating System Market Share Worldwide.

кнопкой любой объект – как предназначенные для этого элементы интерфейса, так и изображения, текст и прочее.

Интерфейс здесь размечается при помощи JSX-разметки, сходной с XML. JSX – это расширение синтаксиса JavaScript, который как раз и предназначен для отделения логики пользовательского интерфейса от прочего кода. Этот фреймворк, однако, не использует визуального редактора интерфейса, представляя разработчику возможность верстать приложение только при помощи разметки программным кодом [3].

Для удобства отладки и создания пользовательских интерфейсов есть так называемый «hot-reload». Эта функция позволяет посмотреть изменения в интерфейсе или функционале приложения при изменении кода без перезапуска отладчика. Таким образом уменьшается время, затрачиваемое на отладку, поскольку при внесённых изменениях заново компилируется не всё приложение, а только изменённые модули.

Поскольку фреймворк существует уже несколько лет, большим плюсом является развитое сообщество разработчиков, использующих этот инструмент. Для React Native существует большое количество проработанных и проверенных временем библиотек и модулей как от компании Facebook, так и от сторонних разработчиков. Также Facebook предоставляет документацию по фреймворку, где описаны принципы его работы и встроенные компоненты. Есть базы знаний, в которых можно найти решения встречающихся проблем, подробно узнать о нюансах разработки.

Производительность приложений на React Native сильно зависит от используемых компонент. Например, приложение будет стабильно работать с частотой 60 кадров в секунду при прокрутке длинных списков, однако частота кадров будет падать при нажатиях на кнопки, набор текста в полях [4]. Это негативно отражается на визуальном впечатлении пользователя от приложения, особенно в нынешней ситуации с распространением дисплеев с частотой 120 Гц и всеобщей тенденции в дизайне к плавным анимациям.

Также к проблемам производительности можно отнести общую нестабильность фреймворка. Зачастую приложение может перестать компилироваться из-за обновлений редактора кода, интегрированных сред разработки, ядра React Native, npm-модулей. Также иногда оказываются нестабильны модули сторонних разработчиков, проблемы с которыми могут появиться через некоторое время после начала разработки приложения, что в некоторых случаях сводит на нет созданные наработки.

В целом, данный фреймворк подойдёт для создания несложных приложений с функционалом «просмотрщика» – когда требуется получить информацию с сервера и отобразить её в виде списка с возможностью его редактирования [5]. Для таких случаев действительно сильно экономится время по сравнению с нативной разработкой под две платформы – iOS и Android. Однако если потребуется реализовать более сложный функционал, особенно при котором происходит взаимодействие с «железной» частью устройства (камерой, Bluetooth и Wi-Fi модулями и др.) – тогда придётся писать много нативного кода и смысл этого кроссплатформенного решения пропадает.

## Xamarin

Xamarin – это фреймворк, поддерживаемый компанией Microsoft, который позволяет разработчикам создавать кроссплатформенные приложения. Ранее компания Microsoft предлагала фреймворки Xamarin.Android и Xamarin.iOS, которые позиционировались как решения для разработки под соответствующие платформы, альтернативные нативным. Xamarin.Forms объединил эти инструменты и теперь этот фреймворк предоставляет

возможность создавать один проект сразу для двух платформ, в отличие от предшественников. В этом проекте описывается вся логика приложения и его графический интерфейс и затем есть возможность скомпилировать проект в исполняемые файлы для операционных систем Android и iOS.

Xamarin поставляется в комплекте MS Visual Studio, что предоставляет возможность использовать богатый набор инструментов этой интегрированной среды разработки. Xamarin.Forms интегрирован с другими продуктами Microsoft, например, со службами Microsoft Azure, что позволяет использовать облачные хранилище файлов и базы данных.

В данном фреймворке используется язык программирования C#, что характерно для продукта компании Microsoft. Это сильно типизированный язык программирования, что сокращает количество непредсказуемых ситуаций при работе с ним. Также C# обладает различными вещами, которые упрощают разработку, такими как, например, лямбда-функции, LINQ, асинхронное программирование [6].

Xamarin.Forms предоставляет собственный набор элементов графического интерфейса. С его помощью можно создать единый интерфейс, который будет выглядеть одинаково на iOS и Android. Это будет являться плюсом, если необходимо реализовать, например, приложение с корпоративным дизайном. Также при помощи инструментов Xamarin.iOS и Xamarin.Android есть возможность использовать нативные элементы, которые обёрнуты в код C# [7]. Это позволяет создавать приложения, дизайн которых соответствует гайдлайнами платформ. Преимуществом этих элементов является повышенная производительность по сравнению с элементами из набора Xamarin.Forms. Есть возможность писать часть кода на нативных языках платформ и вызывать его из кода C#. Для этого, конечно, потребуется иметь хотя бы базовые знания о том, каким образом работают нативные платформы, а также на минимальном уровне владеть нативными языками программирования для них.

Для Xamarin есть ряд библиотек, которые включают классы, поддерживаемые конкретной операционной системой. Это позволяет получать широкий доступ почти к каждой «железной» части смартфона на этих платформах. Также данный фреймворк поддерживает разработку приложений для носимых устройств, работающих на платформах watchOS (например, Apple Watch) и Wear OS (например, Oppo Watch). Это позволяет создавать связанные приложения – такие запускаются на смартфоне и носимом устройстве и взаимодействуют друг с другом [8].

Обычно приложения, созданные при помощи Xamarin, занимают больше места на диске. Например, приложение, которое состоит из одного текстового элемента, в котором будет написано «Hello World» будет весить около 16 мегабайт [8]. Из них только 6 килобайт будет занимать собственно код приложения, остальное – библиотеки Xamarin, стандартное описание интерфейса, системная информация и прочее. Таким образом, приложениям требуется дополнительная оптимизация занимаемого дискового пространства, например, удаление неиспользуемых библиотек и кода.

При интеграции библиотек сторонних разработчиков могут возникать некоторые проблемы. Например, как это бывает со сторонними библиотеками, они в какой-то момент могут перестать обновляться, переставай работать на новых версиях фреймворка. Это в свою очередь может привести к сложностям в развитии разрабатываемого приложения, когда встаёт выбор: отказаться от использования привычной библиотеки или продолжить её использование ценой прекращения перехода на новые версии фреймворка.

Производительность приложений, созданных при помощи фреймворка Xamarin в целом можно охарактеризовать как сбалансированную. Используя этот фреймворк, можно получить достаточную производительность при умеренном бюджете. Приложения в среднем работают

хуже, чем нативные: медленнее запускаются, дольше производят операции с оперативной и постоянной памятью, хуже взаимодействуют с базами данных [2]. Однако это не будет проблемой, если не требуется создавать ресурсоёмких приложений.

## Flutter

Flutter – это инструмент, разрабатываемый компанией Google для создания приложений для Android и iOS с использованием единой кодовой базы. В отличие от других популярных решений, Flutter не является фреймворком: это SDK (software development kit), или набор средств для разработки программного обеспечения, который содержит все необходимое для создания кроссплатформенных приложений. Он включает механизм визуализации, готовые виджеты, API-интерфейсы для тестирования и интеграции, а также инструменты для работы с командной строкой.

Для разработки здесь используется язык программирования Dart, разработанный компанией Google. Он позиционируется как альтернатива JavaScript, развивая положительные качества и стараясь избегать недостатков последнего. Разработчик этого языка утверждает, что Dart оптимизирован для создания пользовательских интерфейсов, в том числе для кроссплатформенных приложений, поскольку позволяет хранить информацию о настройках интерфейса отдельно для каждой платформы. Также этот язык полностью поддерживает концепцию асинхронного программирования, позволяя использовать «async-await» функции при разработке приложений[3].

Особенностью разработки приложений с помощью Flutter является то, каким образом в нём создаются приложения и их интерфейс. Здесь любой объект является виджетом, от кнопки до отступа или шрифта [9]. Виджеты можно комбинировать для создания интерфейса. Работа с виджетами ничем не ограничена – для разработки можно использовать как готовые решения, созданные командой разработчиков Flutter или сторонними программистами, так и самому создать виджет с нуля, описав его функционал и внешний вид. Виджеты здесь организованы в виде дерева – это удобно для рендеринга, но также может привести к путанице в структуре разрабатываемого продукта. Большое приложение может состоять из более чем десяти уровней вложенного кода, поэтому, в таком случае, необходимо тщательно планировать структуру заранее. Flutter предоставляет виджеты, которые идеально соответствуют дизайнерским гайдлайнам для iOS и Android, что как раз и позволяет создавать приложения, идентичные нативным в отношении внешнего вида [9].

При разработке кроссплатформенных приложений обычно происходит разделение это процесса на создание интерфейса и написание кода для реализации функционала. Макет интерфейса может быть описан, например, в XML, и уже в коде происходит связывание элементов интерфейса и функционала. Поскольку всё во Flutter являются виджетами, в них одновременно описывается и внешний вид, и функционал объекта. Таким образом, здесь не происходит разделение процесса создания приложения на разработку интерфейса и написание кода для функционала [10].

[2] https://www.altexsoft.com/blog/engineering/performance-comparison-xamarin-forms-xamarin-ios-xamarin-android-vs-android-and-ios-native-applications/ – Performance Comparison: Xamarin.Forms, Xamarin.iOS, Xamarin.Android vs Android and iOS Native Applications.

[3] https://dart.dev – Dart programming language.

Flutter поддерживает «hot-reload». Как уже описывалось ранее, это позволяет при изменении кода сразу же видеть изменения в приложении в окне эмулятора. «Hot-reload» повышает продуктивность программистов, помогает быстро и удобно делать правки кода.

Поскольку это относительно новый инструмент (первая стабильная версия вышла в декабре 2018 года), сообщество пока что меньше, чем у других популярных инструментов кроссплатформенной разработки. Это означает, что сторонних библиотек будет меньше по сравнению с конкурентными продуктами. Из-за того, что сообщество относительно маленькое, могут появиться проблемы с поиском информации об ошибках и нюансах, возникающих при разработке приложений с помощью Flutter.

Так как Flutter разрабатывается компанией Google, которая также является создателем ОС Android, возникает вопрос о возможном конфликте интересов. Google выгодно, чтобы с помощью их инструмента было удобно разрабатывать приложения для Android, и не появятся ли проблемы в разработкой для платформы от Apple? Однажды произошёл довольно спорный инцидент: одно из крупных обновлений Flutter Release Preview 2 включало виджеты, которые внешне были идентичны элементам интерфейса iOS. Однако, они были актуальны для iOS 10, когда в то же время уже несколько месяцев была доступная новая версия этой операционной системы под номером 11.[4]

Данный инструмент уже используется для создания приложений, например, такими компаниями как «The New York Times», «Square», а также сама компания Google использовала его чтобы создать приложение Google Assistant[5]. Flutter отлично подходит для создания приложений, которые взаимодействуют с информацией на серверах по API и выступают в качестве front-end, но не подходят для создания приложений, завязанных на работе с железом и с дополненной реальностью: для этого придётся использовать большое количество нативного кода. Для создания игр данный инструмент также не подходит ввиду недостаточной производительности по сравнению с нативными инструментами разработки [11].

## Заключение

Инструменты для кроссплатформенной разработки являются отличным решением в том случае, если необходимо создать приложение, не требующее высокой производительности: отображение информации, получаемых с сервером, простые локальные обработчики информации, как, например, приложения для социальных сетей, интернет-магазинов, приложения-органайзеры. Для создания требовательных к вычислительным ресурсам приложений эти инструменты не подходят.

Использование инструментов кроссплатформенной разработки позволяет сократить количество человеко-часов, затрачиваемых на разработку приложений и как следствие, уменьшить финансовые и временные издержки, свести к минимуму отличия в функционале и графическом дизайне между версиями для разных операционных систем. Однако здесь имеются некоторые недостатки, к ним можно отнести использование не самых оптимальных для каждой операционной системы решений (что негативно влияет на скорость работы), низкую по сравнению с нативными инструментами производительность, а также в некоторых случаях несоответствие дизайнерским парадигмам платформ.

[4] https://developers.googleblog.com/2018/09/flutter-release-preview-2-pixel-perfect.html – Flutter Release Preview 2: Pixel-Perfect on iOS.

[5] https://flutter.dev/showcase – Showcase – Flutter.

## ЛИТЕРАТУРА

1. Sahar, C. Clayton iOS 13 Programming for Beginners: Get started with building iOS apps with Swift 5 and Xcode 11. – 4th edition, Packt Publishing, 2020. – 822 p.
2. S. Lewis Native Mobile Development: A Cross-Reference for iOS and Android. O'Reilly Media, 2019. – 394 p.
3. D. Abbott, H. Djirdeh, A. Accomazzo Fullstack React Native: Create beautiful mobile apps with JavaScript and React Native. Independently published, 2019. – 688 p.
4. Adam D. Scott JavaScript Everywhere: Building Cross-Platform Applications with GraphQL, React, React Native, and Electron. O'Reilly Media, 2020. – 344 p.
5. Bonnie Eisenman Learning React Native: Building Native Mobile Apps with JavaScript. – 2nd edition, O'Reilly Media, 2017. – 242 p.
6. D. Hindrikes Xamarin.Forms Projects: Build multiplatform mobile apps and a game from scratch using C# and Visual Studio 2019. – 2nd edition, Packt Publishing, 2020. – 504 p.
7. J. Bennett Xamarin in Action: Creating native cross–platform mobile apps. Manning Publications, 2018. – 608 p.
8. Charles Petzold Creating Mobile Apps with Xamarin.Forms. Microsoft Press, 2015. – 1199 p.
9. Eric Windmill. Flutter in Action. – 1st edition, Manning Publications, 2019. – 368 p.
10. Carmine Zaccagnino Programming Flutter: Native, Cross-Platform Apps the Easy Way (The Pragmatic Programmers). – 1st edition, Pragmatic Bookshelf, 2020. – 370 p.
11. R. Payne Beginning App Development with Flutter: Create Cross-Platform Mobile Apps. Apress, 2019. – 334 p.

**Nedyak Anton Vitalievich**
Far Eastern federal university, Vladivostok, Russia
E-mail: nedyaq@gmail.com

**Rudzeyt Oleg Urievich**
Far Eastern federal university, Vladivostok, Russia
E-mail: rudzeyt18@mail.ru

**Zainetdinov Artem Rishatovich**
Far Eastern federal university, Vladivostok, Russia
E-mail: zainet.ar@gmail.com

**Ragulin Petr Grigorievich**
Far Eastern federal university, Vladivostok, Russia
E-mail: ragulin_p@mail.ru

# Mobile cross-platform app development tools

**Abstract.** This article discusses the existing popular tools for mobile cross-platform application development. It is contrasted with the so – called native application development-when applications are created using the tools provided by the development companies of the respective platforms. Google provides a tool like Android Studio for its Android mobile operating system. The main programming languages used to create applications for this platform are Java and Kotlin. Apple, in turn, offers developers such an integrated development environment as XCode, in which development is carried out using the Swift programming language. The authors reviewed some of the most popular tools for cross-platform development, such as React Native, Xamarin and Flutter. React Native is a product developed by the company Facebook. It inherits the main technological aspects from another framework from Facebook-React. Programming here is conducted in the JavaScript language. React Native is suitable for web developers who have worked with React before and now want to try their hand at developing mobile applications. Xamarin is a framework supported by Microsoft. It provides tools for creating cross-platform applications for the iOS and Android mobile operating systems. This framework is bundled with Microsoft Visual Studio as a downloadable component. What is typical for a Microsoft product, here the C# programming language is used for development. This is a significant advantage of the Xamarin framework: during the creation of a mobile application, developers can use all the important and convenient features of this programming language. Flutter is a tool developed by Google to create apps for Android and iOS using a single code base. Unlike other popular solutions, Flutter is not a framework: it is a set of software development tools that contains everything you need to create cross-platform applications. It includes a visualization engine, ready-made widgets, and tools for working with the command line. The main programming language here is Dart. In the course of studying these tools, the authors made the following conclusions: tools for cross-platform development are an excellent solution if you need to create an application that does not require high performance: displaying information received from the server, simple local information processors, such as applications for social networks, online stores, organizer applications. These tools are not suitable for creating applications that require computing resources.

**Keywords:** cross-platform development; android; iOS; react native; xamarin; flutter; native development; mobile operating system