РАЗРАБОТКА КРОССПЛАТФОРМЕННЫХ ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ KOTLIN MULTIPLATFORM

Вережников Д.В., студент, e-mail:denchic860@gmail.com,
Петрик Е.А., доцент кафедры ПИ,e-mail: petrik.ea@mail.ru,
ЮЗГУ, г. Курск, Российская Федерация

Аннотация. В статье рассмотрена активно развивающаяся технология, позволяющая разрабатывать кроссплатформенные приложения на языке Kotlin.Отмечены преимущества использования данного подхода в проектах.

Ключевые слова: мобильные приложения, десктопные приложения, кроссплатформенная разработка, нативная разработка, kotlin multiplatform, compose.

Введение. Разработка мобильных приложений на сегодняшний день является одним из самых актуальных направлений в IT-индустрии, количество пользователей мобильных устройств растет с каждым годом, а соответственно и спрос на качественные мобильные приложения. Мобильные приложения предоставляют пользователям удобный и быстрый доступ к информации и сервисам, которые ранее были доступны только на компьютерах. Они решают широкий спектр задач: от поиска требуемой информации до онлайн-покупок. Разработка мобильного приложения выгодна для бизнеса, который может использовать его для продвижения своих товаров и услуг, улучшения взаимодействия с клиентами и повышения лояльности.Сегодня разработка приложений охватывает такие мобильные платформы, как iOS иAndroid, на второй план уходят десктопные приложения. По данным портала statista.com [5], количество мобильных приложений, размещенных в магазинах приложений, таких как Google Play,App Store и Amazon Appstore, экспоненциально выросло почти до 6 миллионов, см.рисунок 1.

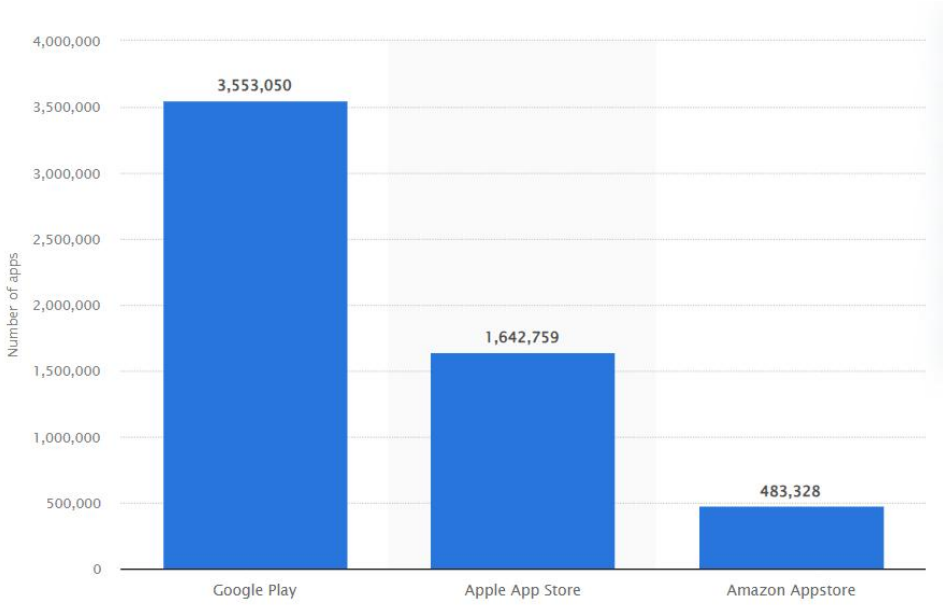

Рисунок 1 – Количество мобильных приложений в ведущихмагазинах

При таком постоянно увеличивающимся количестве конкурентов у компаний возникает вопрос о выборе наиболее подходящей платформы исходя из целей продукта и бюджета. Для разработки приложения под нескольких платформ есть два способа: создание нескольких нативных приложений для каждой платформы или одного – кроссплатформенного.

Нативные приложения пишутся на «родном» для платформы языке и имеют доступ ко всем «родным» функциями ОС, что дает ему большую производительность в сравнении с другими типами приложений. Несмотря на свои достоинства, при поддержке нескольких платформ, нативная разработка оказывается затратной, она требует больше времени и финансовых затрат для заказчика. Альтернативой являются кроссплатформенные приложения, разрабатываемые для нескольких платформ, что существенно экономит время и затраты на разработку. Преимущества кроссплатформенных приложений:

1) универсальность: приложение может работать на различных платформах, используя одинаковый код;
2) экономия: разработка кроссплатформенного приложения дешевле и быстрее, чем нативного приложения для каждой платформы;
3) удобство поддержки: поскольку приложение использует одинаковый код, его поддержка и обновление проще и быстрее.

Несмотря на преимущества кроссплатформенности, зачастую подобные приложения разрабатываются без учета особенностей выбранных платформ, предоставляя одинаковый пользовательский интерфейс для всех. Если потратить время на создание дизайна под все платформы, теряется основное преимущество кроссплатформенной разработки — скорость выхода продукта на рынок.

Недостатки кроссплатформенных приложений:

1) ограниченные возможности из-за абстрагирования от платформы;
2) низкая производительность в сравнении с нативными приложениями;
3) зависимость от кроссплатформенных фреймворков и технологий.

Кроссплатформенные фреймворки добавляют дополнительный слой абстракции между приложением и ОС, что приводит кнеэффективному использованию ресурсов устройства, таким как процессор и память. Такие кроссплатформенные фреймворки, как ReactNative или Xamarin, используют интерпретацию кода, что замедляет работу приложения.

Материалы и методы исследования. Для демонстрации решения проблем кроссплатформенной разработки была рассмотрена технология KotlinMultiplatform (KMP).Данная технология призванаупростить разработку кроссплатформенных приложений, сохраняя гибкость и преимущества нативной разработки. Основная идея заключается в переиспользовании повторяющегося кода между платформами, но в отличие от других технологий, KMP позволяет писать нативный код[2]. Принцип разделения кода между платформами представлен на рисунке 2.

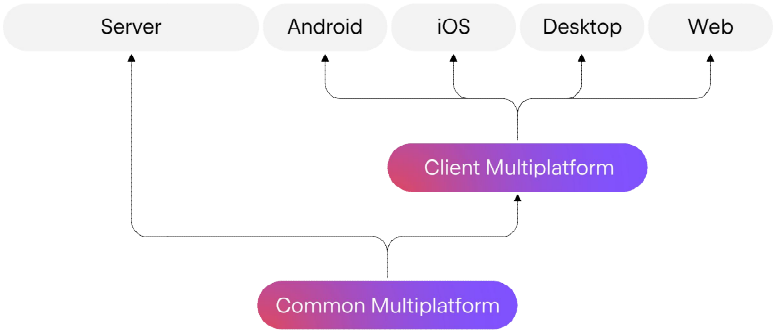

Рисунок 2 – Разделение общего кода между платформами

Мультиплатформенность реализуется с помощью механизма, напоминающий полиморфизм, но на уровне платформ: expect – actual. Expect-функция или класс не имеет реализации и является интерфейсом, который обязательно должен быть реализован отдельно на каждой платформе в виде actual-функции или класса. Нельзя выборочно реализовать expect-класс / функцию на некоторых платформах, программа не скомпилируется, это поведение аналогично стандартным интерфейсам, требующим полной реализации всех своих функций / переменных, см рисунок 4.

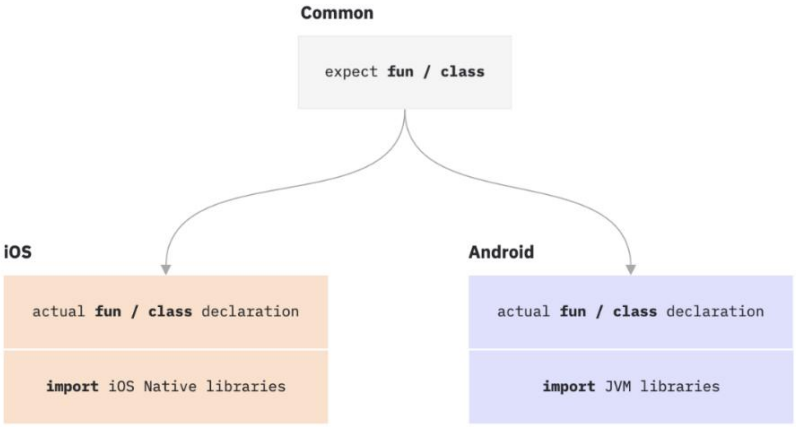

Рисунок 3 – Механизм «expect / actual»

Таргеты. Таргет– это часть сборки, которая собирает и упаковывает код приложения для конкретной платформы, например macOS, iOS или Android. B Kotlinecть 4 основных таргета:

- JVM: код, исполняемыйна виртуальной машинеJava. Есть возможность вызывать Java-код из Kotlinи наоборот;
- Android: код, выполняемый на OCAndroid, совместимый с Java;
- JS: код, транслируемый в JavaScript;
- Native: код, компилируемый в нативные двоичные файлы.

Kotlin Multiplatform поддерживает иерархичную структуру, благодаря которой возможно организовать иерархию промежуточных наборов исходных кодов для совместного использования некоторыми поддерживаемыми целевыми платформами. На рисунке 6 изображен пример использования иерархической структуры в приложении, имеющем десктопную, браузерную и JVM платформы, где есть код, связанный только с десктопными платформами,

который вынесен в отдельный модуль – desktopMain, от которого зависятнижестоящие модули конкретных десктопных платформ (Windows, macOS, Linux).

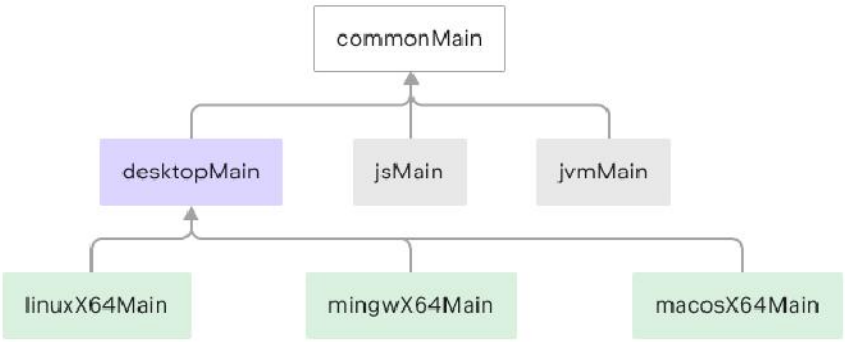

Рисунок 4 – Промежуточный таргет для таргетов десктопных платформ

Несмотря на новизну рассматриваемой в статье технологии у нее уже сформировалась экосистема из различных фреймворков, библиотек и прочих инструментов, см рисунок 7.

Рисунок 5 – Крупные библиотеки и инструменты Kotlin Multiplatform

Мультиплатформенные библиотеки. При выборе мультиплатформенной библиотеки нужно учитывать ее поддерживаемые таргеты. Если в приложении есть таргеты, которые не поддерживает библиотека, ее использование возможно лишь частично. Для создания мультиплатформенной библиотеки нужно настроить ее таргеты. При разработке необходимо определить классы и интерфейсыв общем коде. Следующим шагом будет создание реализации общего кода под каждую платформу.

Тестирование и публикация мультиплатформенного проекта. Вместо написания тестов для каждой платформы отдельно, KMP позволяет написать

общие тесты, выполняемые на всех платформах. Тесты для определенных таргетов будут находиться в модулях, оканчивающихся на Test (jvmTest). Библиотеку можно опубликовать в Maven Central и использовать в других проектах. Чтобы использовать KMP-библиотеку, необязательно использовать KMP в проекте, достаточно поддерживать хотя бы один из таргетов библиотеки.

Результаты и выводы. Была рассмотрена технология, объединяющая преимущества кроссплатформенной и нативной разработки, ускоряющая создание программного продукта. Можно сделать вывод, что кроссплатформенная разработка продолжает набирать обороты. Кроссплатформенные технологии, стремящиеся сохранить доступ к целевым платформам, дают возможность разрабатыватьудобные и производительные кроссплатформенные приложения, учитывая особенности каждой платформы.

Литература

1. Atomic Kotlin, Bruce Eckel, Svetlana Isakova. 2021 ISBN: 9780981872551
2. Kotlin Multiplatform[Электронный ресурс]. Режим доступа: https://kotlinlang.org/docs/multiplatform.html (дата обращения: 15.09.2023)
3. ComposeMultiplatform[Электронный ресурс]. Режим доступа:https://www.jetbrains.com/lp/compose-multiplatform (датаобращения: 17.09.2023)
4. Ktor | Документация [Электронный ресурс]. Режим доступа: https://ktor.io/docs/welcome.html (дата обращения: 09.09.2023)
5. Статистика по количеству приложенийв ведущих магазинах приложений на 3-й квартал 2022 года [Электронный ресурс]. Режим доступа:https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/ (дата обращения: 11.09.2023)