ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ / INFORMATICS AND INFORMATION PROCESSES

DOI: https://doi.org/10.23670/IRJ.2023.132.92

СПОСОБЫ ЛОКАЛИЗАЦИИ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ ДЛЯ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ KOTLIN, НА ПРИМЕРЕ QIZ-APPLICATION

Научная статья

Блинова А.В.[1, *]

[1] ORCID : 0009-0003-2025-4753;

[1] Кубанский Государственный Университет, Краснодар, Российская Федерация

* Корреспондирующий автор (nastya.com107[at]gmail.com)

Аннотация

Благодаря быстрому развитию ИТ-сервисов и не менее быстрому развитию мобильных приложений, все чаще разработчики сталкиваются с ситуацией, когда мобильное приложение должно работать на территории сразу нескольких стран. Часто с этим возникают трудности. Как правило, разные страны имеют разные языковые и культурные особенности, правила правописания, часовой пояс. В этой статье представлен разработанный способ локализации мобильного приложения для операционной системой Android, с использованием классов, интерфейсов и объектов, написанных на языке программирования Kotlin и приведен способ локализации с применением строковых ресурсов. Оба способа продемонстрированы на примере разработанного QIZ-application. Описанные способы позволят локализировать мобильное приложение, чтобы сделать его многонациональным и многоязычным.

Ключевые слова: Android, Android studio, Kotlin, XML, локализация, Qiz-application.

WAYS OF LOCALIZING MOBILE APPLICATIONS FOR THE ANDROID OPERATING SYSTEM IN THE KOTLIN PROGRAMMING LANGUAGE, USING QIZ-APPLICATION AS AN EXAMPLE

Research article

Blinova A.V.[1, *]

[1] ORCID : 0009-0003-2025-4753;

[1] Kuban State University, Krasnodar, Russian Federation

* Corresponding author (nastya.com107[at]gmail.com)

Abstract

Due to the rapid development of IT services and the equally rapid development of mobile apps, developers are increasingly faced with the situation where a mobile app needs to work across several countries at once. This is often a challenge. As a rule, different countries have different language and cultural specifics, spelling rules, time zones. This article presents the developed way of localization of a mobile application for Android operating system, using classes, interfaces and objects written in Kotlin programming language and the way of localization using string resources. Both methods are demonstrated by the example of the developed QIZ-application. The described methods will localize the mobile application to make it multinational and multilingual.

Keywords: Android, Android studio, Kotlin, XML, localization, Qiz-application.

Введение

На сегодняшний день все больше разработчиков нацелены на интегрирование своих приложений по всему миру, в связи с чем, необходимость локализации только растет. Выбранная тема является актуальной еще и по причине того, что строковые ресурсы не используются повсеместно (во всех языках программирования), а реализация при помощи объектов – открывает широкие возможности на любом из них.

Цель данной работы заключается в разработке и применении нестандартного, уникального и более понятного для чтения метода локализации приложения в среде разработки Android Studio на языке программирования Kotlin.

Достижение цели было осуществлено путем решения следующих задач:

1. Анализа стандартного метода локализации с использованием Open Translations Editor;
2. Описания локализации с применением классов и интерфейсов;
3. Сравнения и выявления наиболее удобного и корректного метода локализации приложения.

Язык программирования Kotlin, который использовался для локализации Qiz-application является языком среднего уровня [1, С. 30-36]. Ему присущи ограничения на прямой доступ к памяти, сильная статическая типизация, абстракция и наличие промежуточной среды выполнения [2, С. 115-118]. Философия языка Kotlin твердо опирается на потребности промышленного программирования – быстрая разработка, надежность [3], [4, С. 39-41].

В данной работе использовалась среда разработки Android Studio [5] и установленный в нее эмулятор мобильного устройства, для проверки работы приложения в режиме реального времени. Примененная среда поддерживает несколько языков программирования.

Как было описано ранее, локализация приложения важна и необходима для его использования в нескольких странах. Для увеличения числа пользователей мобильного приложения и масштабирования его на мировом рынке важно программно прописать его локализацию [6, С. 496-502].

Основные результаты

В результате была реализована программная локализации мобильного приложения, с применением классов, интерфейсов и объектов, разработанных по основным принципам языка программирования Kotlin [7, С. 314-508]. Перечисленные компоненты написанные на Kotlin были разработаны и созданы следующие объекты:

1. interface Quiz, у которого в качестве параметров был объявлен список вопросов:

```
interface Quiz {
val questions: List<Question>
}
```

2. class Question, в качестве параметров прописали список вопросов, список ответов и сам вопрос в строковом представлении:

```
class Question(
val question:String,
val answers:List<String>,
val feedback:List<String>,
)
```

3. enum class Locale, создан для перечисления всех языков, которые поддерживаются в разработанном мобильном приложении:

```
enum class Locale {
En, Ru
}
```

4. object QuizStorage, в котором была объявлена функция получения вопросов, ответов и фидбэка:

```
private val quizEn = object : Quiz {
override val questions: List<Question> = listOf(
Question(
question = "How are you feeling?",
answers = listOf(
"Bad",
"Normal",
"Good",
"Wonderful",
),
feedback = listOf(
"You are like a squeezed lemon today.",
"You are not much, but it is been worse.",
"You are in a good mood today.",
"You are in a great mood.",
),
) ....
)
}
```

исходя из выбранного пользователем языка для приложения:

```
fun getQuiz(locale: Locale): Quiz = when (locale) {
Locale.En -> quizEn
Locale.Ru -> quizRu
}
```

Самому объекту был прописан контент на русском и на английском языке, который демонстрировался пользователю, в зависимости от его выбора.

5. После выбора нужного языка происходило его получение:

```
Locale.getDefault().country.toString().
```

6. Далее был прописан код таким образом, чтобы по результату полученного ответа вызывалась соответствующая версия компонентов в приложении:

```
private val questionsList = if
(Locale.getDefault().country.toString() == "RU"){
QuizStorage.getQuiz(QuizStorage.Locale.Ru)
} else{
QuizStorage.getQuiz(QuizStorage.Locale.En)
}
```

Весь код структурирован и отвечает всем требованиям наследования и реализации [8, С. 106-119].

Так же в результате разработки мобильного приложения был программно применен существующий способ локализации приложения. Его суть заключается в создании файлов со строковыми ресурсами под нужный язык с использованием Open Translations Editor [9, С. 34], [10, С. 404]. В данном варианте есть некоторая особенность – это обязательное создание файла, который будет использован по умолчанию (в разработанном приложении это файл на английском языке), на случай если программа не найдет в своих файлах нужного языка для пользователя. В таком случае программой будет использован файл по умолчанию.

Частичный листинг строкового ресурса для английской локализации приложения, выбранной по умолчанию:

```
<resources xmlns:tools="http://schemas.android.com/tools">
```

```
<string name="app_name">HAU</string>
<string name="action_settings">Settings</string>
<!-- Strings used for fragments for navigation -->
<string name="first_fragment_label">First Fragment</string>
<string name="second_fragment_label">Second Fragment</string>
<string name="next">Next</string>
<string name="share_thoughts">Share thoughts</string>
<string name="back">back</string>
<string name="forth">forth</string>
<string name="share">share</string> ....
```

Программно в коде на экранах приложения было прописано обращение к строковым ресурсам с использованием, заранее подключенного в файле сборки проекта, расширения на использование binding. Строковые значения были присвоены view-компонентам внутри функции override fun onCreateView(), передав в качестве параметров несколько значений:

```
inflater: LayoutInflater,
container: ViewGroup?,
savedInstanceState: Bundle?
```

Значения view-компонентам, демонстрирующим список вопросов, были присвоены следующим образом:

```
binding.question1.text = resources.getString(R.string.question_1)
binding.question2.text = resources.getString(R.string.question_2)
binding.question3.text = resources.getString(R.string.question_3)
```

Значения строковых ресурсов автоматически используются из файлов, в соответствии с указанным пользователем языком.

Аналогичным образом были присвоены значения для view-компонентов, разработанных для списков вариантов ответов и списка с feedback.

Разработанное приложение использует в своей конфигурации несколько экранов, наследуемых от класса Fragment(). В результате чего, для сохранения данных между экранами использовалась переменная bundle:

```
val bundle = Bundle().apply {
putString("param1", answer1)
putString("param2", answer2)
putString("param3", answer3)
putInt("param4", picture!!)
}
```

На практике был проведен анализ разработанного и существующего метода локализации для мобильных приложений. В результате к преимуществам разработанного метода можно отнести его простоту в понимании, даже для человека, который впервые видит данную структуру, не составит труда понять принцип действия и логику. Ко второму преимуществу можно отнести быстроту отклика при использовании разработки. Опытным путем было определено, что данный метод на основе классов, интерфейсов и объектов при локализации производил более быстрый отклик, что позволяет смартфону осуществлять перевод на язык пользователя бесшовно, то есть без видимых переходов.

К минусам разработанной технологии относится трудозатраты по написанию кода с использованием классов, интерфейсов и объектов. Для локализации с использованием Open Translations Editor достаточно единоразово использовать его в программном коде в то время как для написания вышеописанной методики необходимо около 70 строк кода. В сравнении с выявленными преимуществами разработанного метода, недостатки можно считать незначительными.

Заключение

В процессе разработки мобильного приложения для операционной системы Android, была проведена локализация Qiz-application, с целью применения разработанного ПО на международном рынке. Она производилась при помощи возможностей среды разработки Android studio, с использованием строковых ресурсов и программно, с применением классов, интерфейсов и объектов, написанных на языке программирования Kotlin. После выполнения описанных в статье действий и доработки мобильного приложения под конкретный случай и нужды пользователя, можно запускать мобильное приложения на устройствах в разных странах мира. Оно будет подстраиваться под выбранный пользователем язык и осуществлять плавный переход с одного языка на другой. Стоит упомянуть, что язык программирования, использованный в данной работе считается молодым и существует довольно мало обучающей информации на русском языке для программистов. Следовательно, данная работа помимо практического применения для разработанного Qiz-application, может служить обучающим пособием для русскоговорящих Android-разработчиков, что впоследствии поможет воспитать Android-разработчиков на языке программирования Kotlin и способствовать развитию сферы мобильной разработки в России.

Конфликт интересов

Не указан.

Рецензия

Ильичев В.Ю., Московский государственный технический университет имени Н.Э. Баумана, Калуга, Российская Федерация
DOI: https://doi.org/10.23670/IRJ.2023.132.92.1

Conflict of Interest

None declared.

Review

Ilichev V.Y., Bauman Moscow State Technical University, Kaluga, Russian Federation
DOI: https://doi.org/10.23670/IRJ.2023.132.92.1

Список литературы / References

1. Жемеров Д.Б. Kotlin в действии / Д.Б. Жемеров, С.С. Исакова — М.: МДК-Пресс, 2021. — 402 с.
2. Дарвин Я.Ф. Android. Сборник рецептов. Задачи и решения для разработчиков приложений / Я.Ф. Дарвин. — СПб: ООО «Альфа-книги», 2020. — 768 с.
3. Altynpara E. The Architecture we Use for Android Apps [Electronic source] / E. Altynpara // The Architecture We Use for Android Apps. — 2022. — URL: https://www.cleveroad.com/blog/clean-architecture-android. (accessed: 28.04.23)
4. Блинова А.В. Противостояние языков программирования Kotlin и Java в разработке мобильных приложений. / А.В. Блинова // Актуальные исследования. — 2022. — 37. — с. 38-41. — URL: https://apni.ru/magazine/116 (дата обращения: 27.04.23).
5. Android Studio. Official website of the development environment [Electronic source] // Android Studio. Official website of the development environment. — 2023. — URL: https://developer.android.com/studio. (accessed: 27.04.23)
6. Стюарт К. Android. Программирование для профессионалов / К. Стюарт, Б. Филлипс, К. Марсикано. — СПб: Питер, 2017. — 688 с.
7. Гриффитс Д. Head First Kotlin / Д. Гриффитс, Дэв. Гриффитс. — СПб: Питер, 2022. — 912 с.
8. Медникс З. Программирование под Android / З. Медникс, Л. Дорнин, Б. Мик и др. — СПб: Питер, 2021. — 496 с.
9. Рафгарден Т. Совершенный алгоритм. Основы / Т. Рафгарден. — СПб: Питер, 2023. — 256 с.
10. Макконнелл С. Совершенный код / С. Макконнелл. — СПб: БВХ, 2022. — 896 с.

Список литературы на английском языке / References in English

1. Zhemerov D.B. Kotlin v dejstvii [Kotlin in Action] / D.B. Zhemerov, S.S. Isakova — M.: MDK-Press, 2021. — 402 p. [in Russian]
2. Darwin I.F. Android. Sbornik retseptov. Zadachi i resheniya dlya razrabotchikov prilozhenii [Android. Collection of recipes. Tasks and solutions for application developers] / I.F. Darwin. — SPb: OOO "Alfa-knigi", 2020. — 768 p. [in Russian]
3. Altynpara E. The Architecture we Use for Android Apps [Electronic source] / E. Altynpara // The Architecture We Use for Android Apps. — 2022. — URL: https://www.cleveroad.com/blog/clean-architecture-android. (accessed: 28.04.23)
4. Blinova A.V. Protivostoyanie yazy'kov programmirovaniya Kotlin i Java v razrabotke mobil'ny'x prilozhenij [The Confrontation of the Kotlin and Java Programming Languages in the Development of Mobile Applications]. / A.V. Blinova // Aktual'ny'e issledovaniya [Current Research]. — 2022. — 37. — p. 38-41. — URL: https://apni.ru/magazine/116 (accessed: 27.04.23). [in Russian]
5. Android Studio. Official website of the development environment [Electronic source] // Android Studio. Official website of the development environment. — 2023. — URL: https://developer.android.com/studio. (accessed: 27.04.23)
6. Stuart K. Android. Programmirovanie dlya professionalov [Android Programming: The Big Nerd Ranch Guide] / K. Stuart, B. Phillips, K. Marsicano. — SPb: Piter, 2017. — 688 p. [in Russian]
7. Griffiths D. Head First Kotlin / D. Griffiths, Dav. Griffiths. — SPb: Piter, 2022. — 912 p. [in Russian]
8. Mednieks Z. Programmirovanie pod Android [Programming Android] / Z. Mednieks, L. Dornin, B. Meike et al. — SPb: Piter, 2021. — 496 p. [in Russian]
9. Roughgarden T. Sovershennyj algoritm. Osnovy [Algorithms Illuminated. Part 1: The Basics] / T. Roughgarden. — SPb: Piter, 2023. — 256 p. [in Russian]
10. McConnell S. Sovershennyj kod [Code Complete] / S. McConnell. — SPb: BVH, 2022. — 896 p. [in Russian]