УДК 004.4'2

РАЗРАБОТКА МОБИЛЬНОГО ANDROID-ПРИЛОЖЕНИЯ НА KOTLIN С ИСПОЛЬЗОВАНИЕМ ПАТТЕРНА MVC

Бондарь И.А., к.э.н., доцент, кафедра МСТ ХНУРЭ
Положай А.Р., студент, кафедра МСТ ХНУРЭ

Аннотация. *Рассмотрена современная технология, которая позволяет ускорить процесс разработки мобильных Android-приложений, повысить качество, как программного кода, так и самого приложения, и способствует упрощению последующей модификации приложения.*

Ключевые слова: *ANDROID, JAVA, KOTLIN, MVC, ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ, РАЗРАБОТКА, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ.*

Количество технологий и библиотек, ускоряющих разработку мобильных приложений, увеличивается с каждым днем. Выбор стека технологий и архитектуры для старта разработки или развития уже созданного мобильного проекта становится все сложнее. Это трудоемкий и ответственный процесс, от которого зависит скорость разработки проекта и состав команды разработчиков.

Для уменьшения трудозатрат на разработку сложного программного обеспечения, в частности мобильного приложения, используются простые и лаконичные языки программирования, и готовые унифицированные решения – паттерны проектирования. Такие решения ускоряют разработку, снижают количество ошибок в коде и облегчают коммуникацию между разработчиками.

В 2008 году вышла первая версия операционной системы Android 1.0 (Apple Pie). Основным языком для написания приложения под нее являлся Java 6.

В 2015 году вышла новая версия операционной системы Android 6.0 (Marshmallow). В этой версии все еще используется Java 6 и некоторые функции Java 7, которые недоступны для старых версий Android SDK. Из этого можно сделать вывод, что Java хоть и является основным языком программирования для разработки мобильных приложений под OS Android, но он развивается недостаточно быстро, а процесс внедрения новой версии в платформу Android длится еще дольше.

Компания «JetBrains» в 2010 году начала разработку нового языка программирования, с целью создать выразительный, статистически типизированный язык, который сможет заменить Java, и на котором можно эффективно писать все компоненты современного приложения. Первый стабильный релиз был выпушен в феврале 2016 года – Kotlin (1.0) [1].

Одно из самых заметных отличий Kotlin от Java – это Nullable типы. Необходимость явно делать проверки на null значительно повышает качество кода, но не является гарантией избавления от NullPointerException [2].

Вторая особенность языка Kotlin – обратная совместимость. Kotlin компилируется в байт-код JVM, что позволяет использовать его в одном проекте с

Java, а возможность взаимно использовать классы делают минимальным порог внедрения Kotlin в уже существующий Java-проект.

Еще одной особенностью языка является набор Extension Functions [3] для стандартных Java-коллекций, которые значительно упрощают работу с данными и сокращают количество кода, что в свою очередь облегчает его последующую модификацию и снижает количество ошибок.

В отличие от Java, Kotlin – лаконичный язык и не требует от разработчика задавать тип переменной или константы самостоятельно. Фрагмент кода, демонстрирующий инициализацию переменных на Kotlin, представлен на рис. 1.

```
1          // пример инициализации переменных
2   var userNames = ArrayList<String>()
3   val user = "PASSENGER"
```

Рисунок 1 – Пример инициализации переменных на Kotlin

В большинстве случаев Kotlin сам может определить тип переменной по коду. Также лаконичность языка обеспечивают строковые шаблоны (рис. 2).

```
1          // строковый шаблон
2   val message = "Hello, $name $lastName"
```

Рисунок 2 – Пример использования строкового шаблона на Kotlin

Язык программирования Kotlin является удобным инструментом, который, несмотря на свои недостатки, повышает продуктивность и расширяет возможности разработчиков, при этом позволяя все так же удобно работать с Java-библиотеками. Простота и гибкость языка дают разработчику больше возможностей для написания быстрого и качественного кода.

Унифицированные решения, такие как паттерны проектирования, представляют собой решение проблем проектирования программного обеспечения в рамках некоторого часто возникающего контекста, что в свою очередь делает разработку быстрее и качественнее.

Model-View-Controller (MVC) – это фундаментальный паттерн, который разделяет данные приложения, пользовательский интерфейс и управляющую логику на три компонента: модель, представление и контроллер (рис. 3).

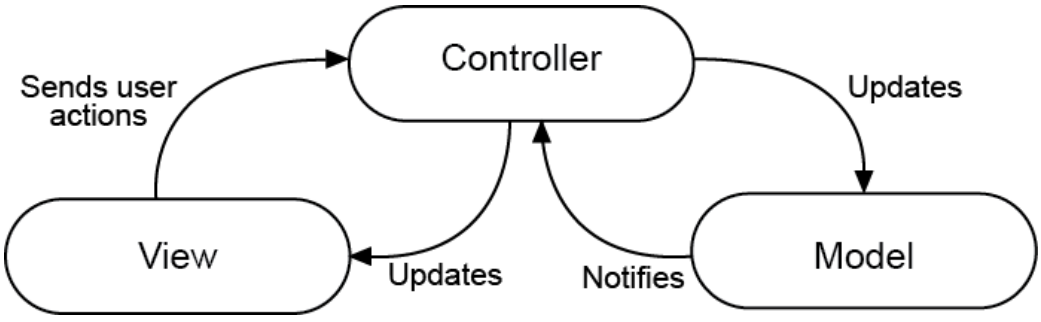

Рисунок 3 – Диаграмма паттерна MVC

Модельный слой описывает данные приложения и определяет логику их обработки и хранения. В Android-разработке модельный слой представляет собой набор классов.

Слоем представления являются XML-разметка и View-компоненты, которые пользователь может увидеть.

Слой управления выступает посредником при взаимодействии объектов слоя представления и модельных объектов. В Android разработке в качестве слоя управления выступает Activity или Fragment. Он контролирует ввод данных пользователем, использует модель и представление для реализации необходимой реакции. Также объекты этого слоя выполняют постановку и согласование задач приложения, управление жизненным циклом других объектов.

Пользователь видит слой представления, производит какие-то действия. Эти действия он перенаправляет слою управления и подписывается на изменения данной модели. Слой управления, в свою очередь, производит определенные действия над модельным слоем. Слой преставления получает последнее состояние модельного слоя и отображает его пользователю.

Очевидное преимущество использования концепции MVC – это четкое разделение логики представления и управляющей логики приложения. Последующая модификация каждого компонента осуществляется независимо.

Помимо изолирования видов от логики приложения, концепция MVC существенно уменьшает сложность больших приложений. Код получается гораздо более структурированным, и, тем самым, облегчается поддержка, тестирование и повторное использование решений.

Литература.

1. Breslav, A. Kotlin 1.0 Released: Pragmatic Language for JVM and Android / A. Breslav. – Режим доступа: https://blog.jetbrains.com/kotlin/2016/02/kotlin-1-0-released-pragmatic-language-for-jvm-and-android. – 11.04.2017. – Загл. с экрана.

2. Oracle. – Режим доступа: https://docs.oracle.com/javase/7/docs/api/java/lang/NullPointerException. – 11.04.2017. – Загл. с экрана.

3. Kotlinlang. – Режим доступа: https://kotlinlang.org/docs/reference/extensions. – 11.04.2017. – Загл. с экрана.

4. Бокарєва, Ю. С., Дейнеко, Ж. В., & Черемський, Р. А. (2016). Інфографіка: сучасний засіб цифрового контенту. In Полиграфические, мультимедийные и web-технологии. Т1. Тез. докл. 1-й Международ. науч.-техн. конф.(16-20 мая 2016)/редкол.: ВФ Ткаченко, ИБ Чеботарева и др.–Харьков: ХНУРЭ, 2016.–208 с. (р. 140).

5. Бокарева, Ю. С., & Дейнеко, Ж. В. (2016). Исследование особенностей плоского и материал-дизайна в UI-интерфейсах.