Ю.Р.Левицька

ДОСЛІДЖЕННЯ JETPACK COMPOSE UI FRAMEWORK ДЛЯ ANDROID. РОЗРОБКА ЗВ'ЯЗКІВ З FRAMEWORK COMPOSE.

Вінницький національний технічний університет

Анотація

В роботі досліджено ряд UI та UX засобів для створення надійного та швидкого додатка на платформі Android з використанням сучасної мови програмування Kotlin. За результатами аналізу обрано оптимальний фреймворк побудови UI та UX для вирішення поставленої задачі.

Ключові слова: Android, kotlin, compose, jetpack.

Abstract

The paper explores a number of UI and UX tools for creating a reliable and fast application on the Android platform using the modern Kotlin programming language. According to the results of the analysis, the optimal framework for building UI and UX was chosen to solve the problem..

Keywords: Android, kotlin, compose, jetpack.

Щороку з'являються нові технології, які полегшують життя розробника. Декларативний інтерфейс користувача став одним із найвизначніших трендів останніх років, оскільки значно підвищує продуктивність розробників зменшує вартість розробки полегшує націлювання на кілька платформ і пристроїв за допомогою одного коду.

Однак Google знадобився деякий час, щоб створити власну декларативну структуру інтерфейсу користувача, перш ніж вони нарешті випустили Jetpack Compose для рідзих програм Android. Jetpack Compose відрізняється від традиційної розробки інтерфейсу користувача.

Починаючи з Java Swing і Win32, переважна більшість інтерфейсу користувача була написана в імперативному стилі. Це також стосується розробки інтерфейсу користувача для Android і iOS. Розробники створювали повнофункціональний інтерфейс користувача, вручну описуючи, як елементи реагують на зміни, і оновлюючи їх пізніше за допомогою сетерів, коли змінюється стан.

Однак React, Flutter, SwiftUI та Jetpack Compose застосували інший підхід. За допомогою цих фреймворків вказуєте, що має представляти інтерфейс користувача, а не як мають бути створені елементи. Частина how залишається для фреймворку, а весь підхід називається декларативним інтерфейсом користувача.Compose є декларативним, і, таким чином, єдиний спосіб оновити

його - це викликати той самий composable з новими аргументами. Ці аргументи є представленнями стану UI. Щоразу, коли стан оновлюється, відбувається перекомпозиція. В результаті такі речі, як TextField, не оновлюються автоматично, як у імперативних представленнях на основі XML. Компоненту має бути чітко вказано новий стан, щоб він відповідним чином оновився.

Компонований, який використовує запам'ятовування для зберігання об'єкта, створює внутрішній стан, роблячи компонуване функцією стану. HelloContent є прикладом компонування з визначенням стану, оскільки він зберігає та змінює свій стан імені всередині. Це може бути корисно в ситуаціях, коли абоненту не потрібно контролювати стан і він може використовувати його, не керуючи станом самостійно. Однак компоновці з внутрішнім станом, як правило, менш придатні для повторного використання і їх важче перевірити.

Коли розробляється багаторазові композиційні файли, вам часто потрібно розкрити як версії того самого компону, що зберігається, так і версії без стану. Версія з визначенням стану зручна для абонентів, які не піклуються про стан, а версія без стану необхідна для абонентів, яким потрібно контролювати або піднімати стан.

Підвищення стану в Compose — це шаблон переміщення стану до виклику компонованого, щоб зробити компонововане без стану. Загальний шаблон для підйому стану в Jetpack Compose полягає в заміні змінної стану двома параметрами подія, яка вимагає зміни значення, де T — запропоноване нове значення. Однак не обмежуєтесь параметрами onValueChange. Якщо більш конкретні події підходять для компонованого, повинні визначити їх за допомогою лямбда, як це робить ExpandingCard з onExpand і onCollapse.

Стан, який піднімається таким чином, має деякі важливі властивості. Єдине джерело істини: змінюючи стан замість того, щоб його дублювати, гарантуємо, що існує лише одне джерело істини. Це допомагає уникнути помилок. Переваги Jetpack Compose UI Framework для Android в вашому додатку:

- висока швидкість роботи програми;
- заощадження пам'яті не мобільному пристрої;
- нова технологія з новими можливостями;
- підтримка корпораціями гігантами;
- заощадження написання коду для UI елементів.

Підсумовуючи, скажемо, що використання Jetpack Compose UI Framework для Android і її окремих функцій дозволяють зосередитися виключно на те, як додаток буде виглядати і наскільки зручним буде його використання. Розглянувши різні засоби для написання та створення інтерактивного та швидкого UI/UX інтерфейсу , було обрано найсучасніший UI Framework, а саме Jetpack Compose. Для вирішення поставленої задачі узгоджено та застосовано

можливості Compose та Fragment. Саме це дозволить додатку бути надзвичайно швидким та надійним у використанні, а також здешевить собівартість та заощадить використаєму память на телефоні.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Jetpack Compose. [Електронний ресурс] : [Веб-сайт] – Електронні дані. - Режим доступу: https://apptractor.ru/info/articles/kontseptsii-jetpack-compose-kotorye-dolzhen-znat-kazhdyy-razrabotchik.html

2. Google выпустила стабильную версию Jetpack Compose 1.0. [Електронний ресурс] : [Веб-сайт] – Електронні дані. - Режим доступу: https://tproger.ru/news/google-nakonec-to-vypustila-jetpack-compose-jetot-instrument-anonsirovali-2-goda-nazad/

3. Jetpack Compose is now 1.0: announcing Android's modern toolkit for building native UI. [Електронний ресурс] : [Веб-сайт] – Електронні дані. - Режим доступу: https://android-developers.googleblog.com/2021/07/jetpack-compose-announcement.html

4. Android Programming In Kotlin: RelativeLayout. [Електронний ресурс] : [Веб-сайт] – Електронні дані. - Режим доступу: http://developer.alexanderklimov.ru/android/layout/ relativelayout.php

5. TableLayout. [Електронний ресурс] : [Веб-сайт] – Електронні дані. - Режим доступу: https://developer.android.com/reference/kotlin/android/widget/TableLayout

Левицька Юлія Русланівна – студентка групи 1КІ-20м, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: yulia@snapp.team

Науковий керівник: Ткаченко Олександр Миколайович – к.т.н, доцент, відповідальний за наукову роботу на кафедрі обчислювальної техніки, Вінницький національний технічний університет, Вінниця, e-mail: alextk1960@gmail.com

Levytska Yuliia - student of the group 1KI-20m, Faculty of Information Technology and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: yulia@snapp.team

Scientific adviser: Alexander Tkachenko - Ph.D., Associate Professor, responsible for scientific work at the Department of Computer Engineering, Vinnitsa National Technical University, Vinnitsa, e-mail: alextk1960@gmail.com