

ПОРІВНЯННЯ АРХІТЕКТУР З ВИКОРИСТАННЯМ ПАТЕРНІВ MVC, MVP І MVVM ПРИ РОЗРОБЦІ ANDROID ДОДАТКУ

Никифоров Д.В., Черних О.П.

*Національний технічний університет
«Харківський політехнічний інститут»,
м. Харків*

Протягом останніх кількох років передовий підхід по розбиттю Android додатків на логічні компоненти еволюціонував. В значній мірі відійшли від монолітного Model View Controller (MVC) шаблону на користь більш модульних моделей, що найкраще тестуються.

Патерні Model View Presenter (MVP) та Model View ViewModel (MVVM) є одними з найбільш широко поширених альтернатив стандартному підходу, але розробники часто сперечаються, який краще підходить до Android [1].

MVC підхід розділяє додаток на макрорівні 3-х наборів обов'язків: даних, представлення та бізнес логіки. Робить велику роботу відділення моделі і представлення. Модель можна легко протестувати, бо вона не прив'язана ні до якого контексту. Але контролер прив'язаний настільки щільно до Android API, що важко виконувати модульне тестування. Контролери тісно пов'язані з представленням. При зміні представлення треба переписувати контролер.

За допомогою MVP зв'язок з представленням може відбуватися без прив'язки його до решти «контролерів». Це набагато чистіше. Легко протестувати блок логіки презенторів, тому що він не прив'язаний до Android API, що також дозволяє працювати з будь-яким іншим представленням. Презентори, так само, як контролери, схильні до збору додаткової бізнес-логіки. В якийсь момент, розробники часто залишаються з великими громіздкими класами, які важко розірвати один від одного [1].

MVVM з прив'язкою даних на Android має переваги раннього тестування і модульності, а також зменшення кількості коду, який ми повинні написати, щоб підключити представлення плюс модель [2]. Модульне тестування стає ще простішим, тому що насправді немає ніякої залежності від представлення.

У розділенні додатку на модульні компоненти патерни MVP і MVVM краще виконують роботу, ніж MVC. Проте вони також додають додаткові складності для розробленого додатку. Для дуже простого додатку з одним або двома екранами, MVC може працювати краще. MVVM з прив'язкою даних є привабливим та робить модель більш реактивною і має менший код. Правильним підходом є також комбінування даних патернів в залежності від поставленої задачі [2].

Література:

1. Калачан А. Архитектура Android-приложений. Правильный путь? – Режим доступа [www. URL: \[https://habrahabr.ru/post/250659/\]\(http://www.habrahabr.ru/post/250659/\)](http://www.habrahabr.ru/post/250659/) – 10.02.2017. 2. Google Data Binding library – Режим доступа URL: <https://developer.android.com/topic/libraries/data-binding/index.html> – 10.02.2017.