

13-14 червня, 2017

ДонНТУ, ФКНТ, корпус 3, ауд. 3.319/1



## Тренінг GameHub

# Розробка ігрових додатків на базі Unity та ОС Android

---

Архітектура і базові принципи проектування ігрових додатків для мобільних пристроїв під керуванням ОС Android



























# Архітектура з використанням RxJava – DataManager

**DataManager** є центральною частиною архітектури. Він використовує оператори RxJava для того, щоб комбінувати, фільтрувати і трансформувати дані, отримані від помічників.

Тобто Activity і фрагменти звільняються від роботи по впорядковуванню даних – DataManager буде виробляти всі потрібні трансформації всередині себе і віддавати дані, готові до відображення.



**Класи-помічники** мають дуже обмежені області відповідальності, і реалізують їх в послідовній манері.

Наприклад, більшість проектів мають класи для доступу до REST API, читання даних з БД або взаємодії з сторонніми SDK.

У різних додатків буде різний набір класів-помічників, наприклад:

- PreferencesHelper – працює з даними в SharedPreferences
- DatabaseHelper – працює з SQLite
- Сервіси Retrofit, що виконують звернення до REST API
- тощо...



1. Observables і оператори з RxJava позбавляють від вкладених функцій зворотного виклику.
2. DataManager бере на себе роботу, яка раніше виконувалася на рівні уявлення, розвантажуючи таким чином Activity і фрагменти.
3. Переміщення частини коду в DataManager і класи-помічники робить юніт-тестування Activity і фрагментів простішим.
4. Чітке розділення відповідальності і виділення DataManager робить всю архітектуру більш дружньою до тестування.
5. Класи-помічники, або DataManager, можуть бути легко підмінені на спеціальні заглушки.



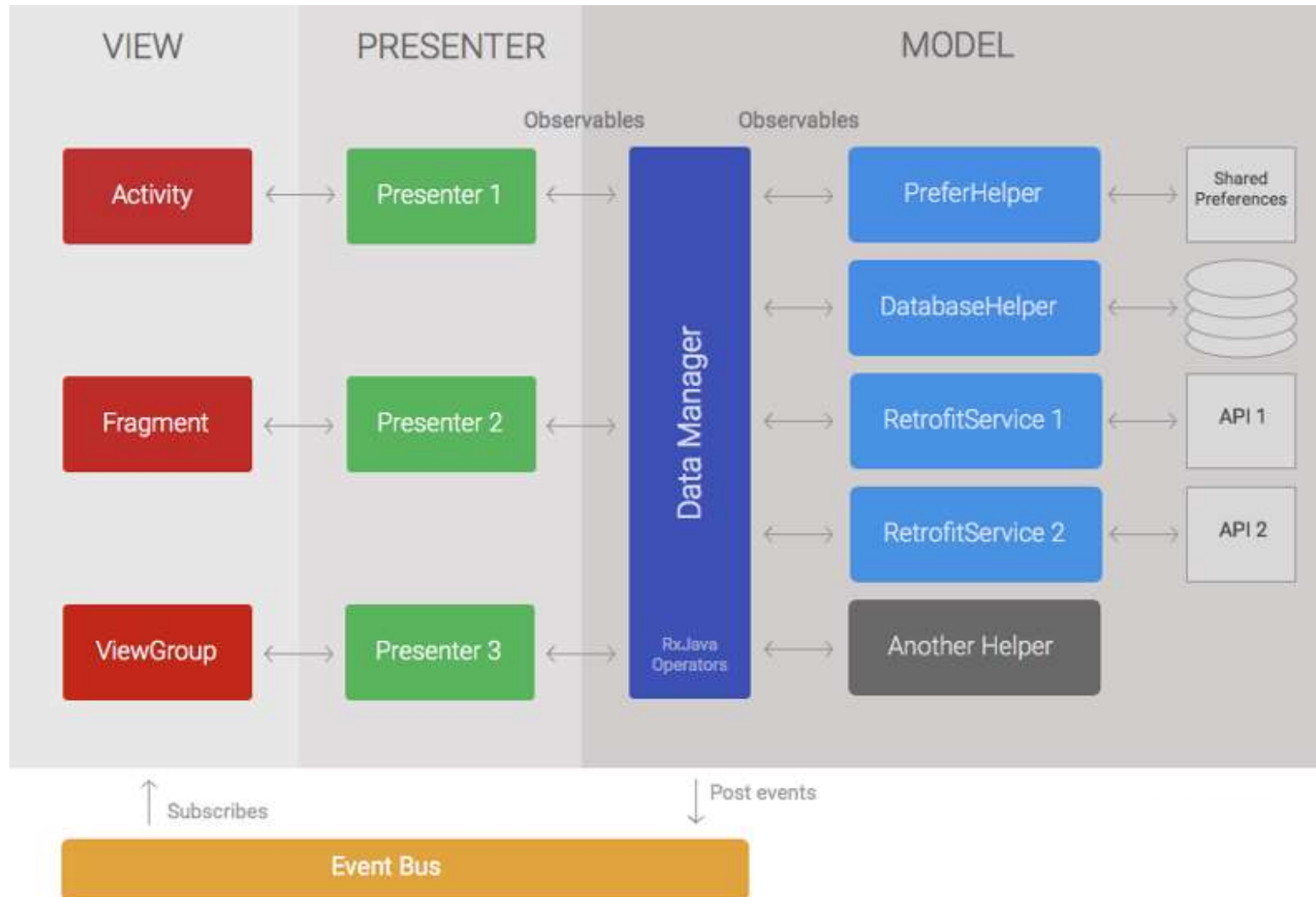
# Перехід до Model View Presenter



Co-funded by the  
Erasmus+ Programme  
of the European Union



G A M E H U B  
University Enterprises Cooperation  
in Game Industry in Ukraine





Presenter'и відповідають за завантаження даних з моделі і виклик відповідних методів на рівні уявлення, коли дані завантажені.

Presenter'и підписуються на Observables, які повертаються DataManager. Отже, вони повинні працювати з такими сутностями як підписки і планувальники. Більш того, вони можуть аналізувати виникаючі помилки, або застосовувати додаткові оператори до потокам даних, якщо необхідно.

Наприклад, якщо потрібно відфільтрувати деякі дані, і цей фільтр швидше за все ніде більше використовуватися не буде, є сенс винести цей фільтр на рівень presenter'a, а не DataManager.