В.І. Драйчук, магістрант, гр. ПІ-43м, ФІКТ
Науковий керівник – д.т.н., проф. А.В. Панішев
Житомирський державний технологічний університет

АКТУАЛЬНІСТЬ ВИКОРИСТАННЯ РЕАКТИВНОГО ПРОГРАМУВАННЯ ПРИ РОЗРОБЦІ ДОДАТКІВ НА ОС ANDROID

Android — одна з найбільш актуальних на сьогодні мобільних операційних систем. Дана операційна система створена компанією Google на базі ядра Linux. Базовим елементом цієї операційної системи є реалізація Dalvik віртуальної машини Java, і все програмне забезпечення і застосування спираються на цю реалізацію Java. Варто визнати, що майбутнє ПК – у портативній, легкій та функціональній техніці – планшетах, електронних книжках, нетбуках і смартфонах, і все це в основному працює саме на операційній системі Android.

В травні 2010 року був досягнутий рубіж у 100 тисяч активацій в день. В грудні 2010 в день активувалося вже 300 тисяч апаратів. У травні на конференції Google I/O була озвучена статистика, згідно з якою щодня активується близько 400 тисяч нових пристроїв на базі платформи Android. В липні 2011 Енді Рубін (Andy Rubin), віце-президент Google, який відповідає за розробку платформи Android, повідомив про подолання нової межі — 500 тисяч активацій на день при зростанні поширення платформи в 4,4% на тиждень. У статистику активацій включена тільки інформація про першу реєстрацію нових пристроїв, що поставляються з набором сервісів Google. Апарати з прошивками без набору застосунків Google, що випускаються деякими азіатськими виробниками, у статистиці не враховані.

Загалом станом на липень 2011 продано вже понад 100 млн Android-пристроїв, випущених 36 виробниками, їх розповсюджували у мережах 215 операторів зв'язку. Всього було продано більш ніж 200 млн Android-пристроїв. Каталог Google Play подолав позначку в 200 тисяч додатків. Всього з Google Play встановлено близько 4.5 мільярдів копій програм. За даними Google на вересень 2012 року активовано 500 мільйонів «Андроїдів». На 84% смартфонів, проданих у третьому кварталі 2014 року, була встановлена операційна система Android. Таким чином розробка додатків для ОС Android на сьогоднішній день є украй актуальною.

Працюючи з Android часто можна бачити, як весь функціональний код розташовується в методах життєвого циклу activity / fragment. Насправді такий підхід має деяке обгрунтування - «методи життєвого циклу» всього лише обробні етапи створення компонента системою і спеціально призначені для наповнення їх кодом. Додавши сюди те, що каркас UI описується через xml-файли, ми вже отримуємо базовий поділ логіки і інтерфейсу. Однак через не зовсім «витончену» структуру життєвого циклу, і різної (хоч і схожої) структури для різних компонентів, ефективно скористатися подібним поділом не завжди буває можливо, що в підсумку виливається в написанні всього коду в onCreate(), що негативно впливає на зрозумілість, прозорість коду і робить майже неможливим його модифікування та супроводження.

В свою чергу RxJava – це нова технологія, яка зараз є однією з найгарячіших тем для обговорення у Android-програмістів. Реактивне програмування — це парадигма програмування, побудована на потоках даних і розповсюдженні змін. Це означає, що у мовах програмування має бути можливість легко виразити статичні чи динамічні потоки даних, а реалізована модель виконання буде автоматично розсилати зміни через потік даних. Відмовостійкість, чутливість, орієнтованість на події і масштабованість – це ті основні принципи реактивного програмування, дотримання яких полегшує роботу з кодом. Саме дотримуючись їх створюється backend великих систем з одночасною підтримкою десятків тисяч з'єднань. Реактивне програмування - програмування з використанням асинхронних потоків даних. У певному сенсі, це не є чимось новим. Типові події кліків є асинхронними потоками подій, над якими і робилися деякі побічні ефекти. Ви можете створювати потоки даних від будь чого, а не тільки від миші і події переміщення курсору. Все що завгодно може бути потоком: змінні, інформація, яку користувач вводить, властивості, структури даних і т.д. Наприклад, канал Twitter буде випромінювати потік даних, таким же чином, що і миша події про переміщення чи клік. Ви можете слухати цей потік і реагувати відповідним чином.

Вивчення можливостей застосування реактивного програмування, а саме RxJava, для написання додатків для ОС Android є важливим оскільки використання даної технології суттєво полегшує написання коду та обробку помилок. Дана технологія надає широкі можливості для фільтрування, обробки результатів викликів, тощо.

Деякі можливості використання RxJava в Android додатках: використання RxJava для спрощення багатопоточних RESTful API викликів; використання RxAndroid для оптимізації роботи з компонентами UI операційної системи Android; спрощена робота з основними потоками додатку. Використання технології RxJava надає безліч переваг при написанні та супроводі додатків, насамперед відмовостійкість: результат виконання всіх операцій завжди передбачуваний. Ми знаємо про всі помилки і потенційно проблемних місцях, які можуть виникнути в коді, і вже обробили їх. Ніяких винятків не буде. Принцип чутливості в дії: з'єднання з базою даних або з сервером не розірветься завдяки таймауту, виклик спробує відновитися, при помилці і, що теж важливо, поверне результат відразу, до кешування. А кешування виконається паралельно обробці результату. Орієнтованість на події: по ходу виконання запиту, ми завжди реагуємо на події - події успішного або неуспішного завершення запиту, подія закінчення і т.д. Код легко розширюється і майже не вимагає змін. Якщо нам необхідно зробити обробку помилки або збереження стеку помилок, можна з легкістю обробити це за допомогою компонентів RxJava. Найбільш доцільним є використання RxJava у наступних випадках: UI події, як переміщення миші, натиснення кнопок; події з WebSocket API; події, такі як зміна властивості, оновлення колекції, успішно пройдена реєстрація, тощо. Не радять використовувати RxJava виключно для перебору колекцій, для цього значно доцільніше використовувати звичайні ітератори. Rx грунтується на шаблоні Observer та системі підпису на події. Ідея Rx в тому, що невідомо, коли послідовність видає значення або закінчується, але ви маєте контроль над тим, коли ви починаєте і припиняєте приймати значення. Базовими будівельними блоками реактивного коду є Observable і Subscribers. Observable - це як потоки, які можуть будь-яким чином отримувати і віддавати дані. Observable є джерелом даних, а Subscriber -

споживачем. Породження даних через Observable завжди відбувається відповідно до одного і того ж самого порядку дій: Observable «випромінює» певну кількість даних (в тому числі, може нічого і не випромінювати), і завершує свою роботу - або успішно, або з помилкою. Для кожного Subscriber, підписаного на Observable, викликається метод Subscriber.onNext() для кожного елемента потоку даних, після якого може бути викликаний як Subscriber.onComplete(), так і Subscriber.onError(). Все це дуже схоже на звичайний патерн «Спостерігач», але є одна важлива відмінність: Observables часто вже не починають породжувати дані до тих пір, поки хто-небудь явно не підписується на них.

Над Observable можна застосовувати оператори, такі як flatmap, filter, zip, merge, cast і т.д. Оператори можуть бути використані в проміжку між Observable і Subscriber для маніпуляції над даними. В RxJava існує дуже багато операторів, тому для початку краще буде зосередитися лише на деяких з них.

- Create - створити Observable з нуля шляхом виклику observer методів;
- From - конвертувати будь-якої інший об'єкт або структуру даних в Observable;
- Map - трансформувати елементи, які випромінює Observable шляхом застосування функції до кожного елементу;
- Timer - створити Observable, який випромінює один елемент після даної затримки;
- Start - створити Observable, який випромінює значення функції;
- Filter - виділити лише ті елементи з Observable, які проходять тест предикат;

Реактивні додатки дійсно є більш стійкими до багів і непередбачуваних помилок, використання технології робить код зрозумілим і більш гнучким. Багато рутинної роботи перекладається на бібліотеку, яка виконує свою роботу краще, ніж базові Android-компоненти. Це дозволяє зосередитися на реалізації речей, які дійсно варто обміркувати.

Реактивние програмування - це трохи інше мислення в порівнянні з традиційною розробкою під Android. Потоки даних, функціональні оператори - ці складні, на перший погляд, речі виявляються набагато простішими, якщо розібратися. Це потужна частинка ФП в ООП, яка робить життя і код простішим, а додаток кращим.

Rx працює відмінно для перевантажених контентом додатків. Але потенціал технології розкривається не лише на стороні клієнта, він прекрасно працює і на стороні сервера і близько до баз даних. Насправді, RxJava є ключовим компонентом на стороні сервера в API компанії Netflix. Rx не являє собою фреймворк, і не обмежується одним конкретним типом додатків або мови. Насправді це парадигма, яку можна використовувати при програмуванні будь-якого керованого подіями програмного забезпечення і яка є актуальною та в цілому позитивно впливає на якість кінцевого коду.