та сім'ї. – 2014. – № 5. – С. 20–21.
3. Закон України про вищу освіту [Електронний ресурс]. – Режим доступу : http://zakon2.rada.gov.ua/laws/show/1556-18.
4. Дочкин С. А. Использование пакетов свободного программного обеспечения в дополнительном профессиональном образовании / С. А. Дочкин // Научное обеспечение системы повышения квалификации кадров. – 2011. – Вып. 2 (7). – С. 46–52.
5. Рудик О. Перспективи платформнонезалежної та Linux-орієнтованої інформатики в Україні [Електронний ресурс] / О. Рудик. – Режим доступу : http://www.kievoit.ippo.kubg.edu.ua/kievoit/c130829/4-Rudyk.html.
6. Карпенко М. Перспективи та можливості впровадження вільного програмного забезпечення в навчальних закладах та державних установах України [Електронний ресурс] / М. Карпенко, М. Кияк. – Режим доступу : http://old.niss.gov.ua/MONITOR/june2009/15.htm.

Розробка Android-додатків на Android Studio під Ubuntu на Kotlin

Яхненко О.В

Полтавський національний педагогічний університет імені В.Г. Короленка, yahnencko.sasha@gmail.com

В доповіді розглядаються рекомендації до впровадження вільного програмного забезпечення в освіті з метою спрямування користувачів, які стали перед вибором засобів розробки. З поетапним аналізом найбільш доцільних та доступних засобів для реалізації.

На сьогоднішній день враховуючи темпи впровадження найрізноманітніших пристроїв в наше життя, одним з найактуальніших методів впровадження ВПЗ є дана сфера. Такий висновок зроблено виходячи з тенденції розвитку технологій, враховуючи частку яка припадає саме на мобільні пристрої [2]. Про це говорить статистика представлена компанією Google, де вказано що кількість інформаційних запитів зроблених з мобільних пристроїв, значно більша ніж з ПЕОМ (статистика на основі 10 країн включно з США та Японією). Таким чином ми стикаємося з вибором між iOS, Android чи Windows Phone. В першу чергу слід зазначити, що для розробки додатків на тій чи іншій платформі бажано мати пристрій де можна буде провести процес відладки. iPhone завжди був одним з найдорожчих смартфонів на ринку. iPhone 5c коштує 250$, не кажучи вже про 6S+ з цінником в 1100$. З Windows Mobile виникають проблеми як в плані наявності ліцензії для розробки додатків на цій платформі, так і подальше просування свого товару на ринок. Масштабами і гнучкості у Android немає конкурентів. Мобільні пристрої на Android робить чи не кожен, хто пов'язаний з ринком смартфонів і тут завжди можна вибрати варіант від «дуже дешево» до «я не маю стільки грошей». А за величезну кількість безкоштовних додатків в PlayMarket Google отримує +5 до карми.

На Android все дуже просто: додаток можна завантажити безпосередньо

з онлайн-магазину, встановити з комп'ютера або скористатися одним із сторонніх магазинів яких існує велика кількість. А ось Apple і Windows навпаки відкрито виступають проти сторонніх магазинів. Тому якщо ви хочете мати простір для маневру, то чаша терезів схиляється в бік Android.

Недовге життя батареї - це головний біль і користувачів, і виробників. Вважається, що у iOS система енергоспоживання більш збалансована, однак Windows і Android виграють за рахунок більш містких акумуляторів, які зараз ставить кожен другий виробник.

Для кастомізації оболонки можна змінити вбудоване програмне забезпечення, встановити лаунчер, змінити екран блокування, тасувати віджети і змінювати розміри ярликів. За статистикою 85% всіх смартфонів, проданих у другому кварталі 2015 року, використовують ОС Android. Щодня в світі активується більше 1,3 млн пристроїв на базі ОС Android, і слід зазначити, що це не лише смартфони, а найрізноманітніші пристрої на кшталт цифрових годинників, білбордів чи навіть “розумного будинку” .Визначившись із платформою постає питання з вибором середовища програмування. В цьому питанні це скоріше вимога часу. Чому ж саме Andoid Studio:

1. Gradle. Надзвичайно потужний інструмент для збору проекту. Найбанальніший приклад використання - це під’єднання бібліотеки прописуванням одного рядка в файлі build.gradle на рівні додатку. До переваг системи слід віднести швидкодію програми яка зростає з кожною новою версією. Постійне оновлення бібліотек та зміна окремих модулів забирає багато часу на пошук помилки в одному з файлів, гнучкість gradle-змінних дасть змогу уникнути такої ситуації. У розроблюваних нами додатках ми часто використовуємо сторонні бібліотеки, що містять різні внутрішні ресурси, які в наших додатках абсолютно не потрібні. Завдяки Android Gradle Plugin ми можемо встановити мови і розширення, які використовуються в додатку, інші будуть вилучені, що дасть змогу зменшити розміри файлів[1].

2. Вбудований SDK. Існує багато різних IDE з уже вбудованим Android SDK, але додані вони туди не так тісно як в студії. Наприклад вам потрібно запустити старий проект, з низьким API level, який за непотрібністю ви просто не завантажили, половина класів заливаються фарбою, кількість помилок в проекті невблаганно росте і ви розумієте, що чогось тут не вистачає, але чого? Якої версії API немає? - потрібно розбиратися. Подібної ситуації у разі використанні студії не виникає. У разі використання будь-якого елементу, якого у вас немає, вона сама визначити де і чого не вистачає і викине віконце з повідомленням. Більш того, всі необхідні елементи можуть бути встановлено за лічені секунди.

3. Зручний інтерфейс. З першого погляду він не має ніякої відмінності від аналогічного в Eclipse, але понатискавши незнайомі кнопки, змінюючи відображення екрану і перегляді доданих нами елементів інтерфейсу, можна відразу зрозуміти, наскільки все стало краще. В одне натискання можна

переглянути відображення нашого екрану на будь-якому пристрої, аж до телевізорів і годинників. Самі елементи інтерфейсу відображаються саме так, як вони будуть виглядати у конкретній версії ОС, на відміну від інших IDE, де віджети відображаються на всіх версіях у вигляді однієї стандартної картинки.

4. Зручний дизайн. В решті-решт це стара добра IDEA, яку знають як свої п’ять пальців добра частина Java розробників у всьому світі [4]. Дуже велику частину простору займають вікна повідомлень, властивостей елементів і самого проекту, але для кожного вікна передбачена функція згортання в одну з частин інтерфейсу. Більш немає потреби, як це було в Eclipse, закривати вікна за непотрібностю, всі ваші вікна будуть знаходиться в одному місці на панелі задач.

Наступним кроком буде вибір операційної системи на якій власне кажучи і буде відбуватись весь процес. З одного боку це швидше питання вподобань смаків чи навіть просто звички, те з чим найчастіше доводилось працювати користувачеві. Але якщо більш детально заглибитись в це питання, то можна побачити очевидні переваги Ubuntu.

Почнемо з того, що практично всі написані для Linux програми абсолютно безкоштовні. Так історично склалося, що програми під Linux поширюються з відкритим вихідним кодом. Це дає змогу запускати програму в будь-якій системі. Важко вимагати гроші за те, що може трохи підправити і поширювати безкоштовно будь-який студент-програміст.

У світі Linux майже не можливо підхопити якийсь вірус. З великою точністю можна вважати, що їх взагалі немає.

В порівнянні з Mac ОС Ubuntu виграє за рахунок того, що працює на будь якому комп’ютері та з будь якою конфігурацією обладнання. До плюсів можна віднести прості автоматичні оновлення, можливість вибору, і інтеграцію програмного забезпечення незалежно від його вихідного коду, і напевно найголовнішим аргументом є ціна, де Ubuntu немає рівних за доступністю. Стосовно Microsoft Windows можна сказати, що Linux працює швидше. По-перше, раз немає вірусів, отже пересічному користувачеві не доводиться встановлювати антивіруси і міжмережеві екрани. І ті, і інші існують в світі Linux, але використовуються переважно на серверах. По-друге, оскільки програми поширюються у вигляді вихідного коду, то під час компіляції можна провести оптимізацію під конкретну систему. Згодом операційна система не буде гальмувати. Програми не вимагають перезавантаження системи після встановлення, оновлення або видалення. Оскільки за ПЗ платити не треба, то розробники не женуться за випуском нових релізів. А отже ніхто не женеться за прикрашанням і не потрібним функціоналом, займаючись натомість оптимізацією програм. Як наслідок, Linux добре працює на старому залізі, а також смартфонах і нетбуках; роздолля для програміста - всі протоколи і бібліотеки прекрасно задокументовані, можна заглянути в код будь-якої програми і дізнатися, як

вона працює. Емуляція віртуальних машин не працює на процесорах від AMD в операційній системі Microsoft Windows, тому що Microsoft лобіює інтереси Intel. На офіційному сайті Android Developers сказано, що скористатися емулятором Android на AMD можна тільки в Unix-сумісній системі. ARM-образи працюють занадто повільно для нормального їх використання. [5]

Переходячи до останнього і найбільш складного етапу, слід зазначити що в сучасному світі є досить велика кількість мов програмування, що спочатку ставить перед складною альтернативою - яку ж мову обрати?Звісно можна скористатись різними рейтингами, статистикою, або не задумуватись над вибором та стати на бік більшості. Але слід дивитись більш прагматично та звернути увагу на нові мови, які розвиваються, що в свою чергу свідчить про те що вони враховують всі недоліки та переваги своїх попередників. Такий чином вибір зупиняється на такій мові як Kotlin.

Дві головні особливості Kotlin, це її простота і повна сумісність з Java. Мова Kotlin створювалась компанією, яка робить дуже багато продуктів мовою Java і яка добре розбирається в сучасних інструментах розробки.

Сумісність з Java. Платформа Java - це перш за все екосистема: крім «офіційних» продуктів компанії Oracle, в неї входить безліч проектів з відкритим кодом: бібліотек і програмних каркасів різного профілю, на базі яких будується величезна кількість додатків. Тому для мови, яка компілює для цієї платформи, дуже важлива сумісність з наявним кодом, який написаний мовою Java. При цьому необхідно, щоб наявні проекти могли переходити на нову мову поступово, тобто не тільки код мовою Kotlin повинен легко викликати код мовою Java, але і навпаки.

Статичні гарантії коректності. Під час компіляції коду з статично типізованою мовою відбувається безліч перевірок, покликаних гарантувати, що ті чи інші помилки не відбудуться під час виконання. Наприклад, компілятор Java гарантує, що об'єкти, на яких викликаються ті або інші методи, «вміють» їх виконувати, тобто що у відповідних класах ці методи реалізовані. На жаль, крім цієї дуже важливої властивості, Java майже нічого не гарантує. Це означає, що успішно скомпільовані програми завершуються з помилками часу виконання (викликають виняткові ситуації). Яскравим прикладом є розіменування нульового посилання, при якому під час виконання викликається виняток типу NullPointerException. Важливою вимогою до нової мови є посилення статичних гарантій. Це дасть змогу виявляти більше помилок на етапі компіляції і, таким чином, скорочувати витрати на тестування.

Швидкість компіляції. Статичні перевірки спрощують програмування, але уповільнюють компіляцію, і тут необхідно домогтися певного балансу. Досвід створення мов з потужною системою типів (найбільш яскравим прикладом є Scala) показує, що такий баланс знайти непросто: компіляція часто стає неприйнятно довгою. Взагалі, така характеристика мови, як час

компіляції проекту, може здатися другорядною, однак в умовах промислової розробки, коли обсяги компільованого коду дуже великі, виявляється, що цей фактор дуже важливий - адже поки код компілюється, програміст часто не може продовжувати роботу. Зокрема, швидка компіляція є одним з важливих переваг Java в порівнянні з C ++ і Kotlin повинен цю перевагу зберегти.

Лаконічність. Відомо, що програмісти часто витрачають більше часу на читання коду, ніж на його написання, тому важливо, щоб конструкції, доступні в мові програмування, давали змогу писати програми коротко і зрозуміло. Java вважається багатослівною, і завдання Kotlin - поліпшити ситуацію в цьому сенсі. На жаль, суворі методи оцінювання мов з точки зору їх лаконічності розвинені досить слабо, але є непрямі критерії; один з них - можливість створення бібліотек, робота з якими близька до використання предметно-орієнтованих мов (Domain-Specific Language, DSL). Для створення таких бібліотек необхідна певна гнучкість синтаксису в поєднанні з конструкціями вищих порядків; найбільш поширені функції вищих порядків, тобто функції, які беруть інші функції в якості параметрів.

Доступність для вивчення. Складні статичні перевірки, гнучкий синтаксис і конструкції вищих порядків ускладнюють мову і утруднюють її вивчення, тому необхідно до певної міри обмежувати набір підтримуваних можливостей, щоб мова була доступна для вивчення. Під час розробки Kotlin враховувався досвід Scala і інших сучасних мов, і занадто складні концепції в мову не вносились.

Інструментальна підтримка. Сучасні програмісти активно використовують різні автоматизовані інструменти, центральне місце серед яких займають інтегровані середовища розробки (Integrated Development Environment, IDE). Десятирічний досвід, накопичений в компанії JetBrains, показує, що певні властивості мови можуть істотно ускладнювати інструментальну підтримку. Під час розробки Kotlin враховано цей факт і створено IDE одночасно з компілятором.[3]

Отже, актуальність цього дослідження визначається потребою спрямування користувача в сфері розвитку вільного програмного забезпечення, враховуючи альтернативність різних методів розробки. В доповіді наведено причини вибору кожного сегменту окремо, їх взаємодію, та продуктивність роботи системи в цілому. Таким чином, кожен, хто усвідомлює перспективність роботи в цій сфері, може стати Android-розробником. В сучасному світі для цього не потрібні дорогі ліцензії чи спеціальне обладнання.

Джерела

1. Gradle: 5 полезностей для разработчика [Електронний ресурс] : Блог компании REDMADROBOT / Хабрахабр . — Режим доступу до блогу: https://habrahabr.ru/company/redmadrobot/blog/271269/
2. Bosomworth D. Mobile Marketing Statistics compilation [Електронний ресурс] : Smart insights. — Режим доступу до сайту: http://www.smartinsights.com/mobile-

marketing/mobile-marketing-analytics/.
3. Leiva A. Kotlin for Android Developers [Електронний ресурс] :Leanpub: Publish Early, Publish Often. — Режим доступу до сайту: https://leanpub.com/kotlin-for-android-developers
4. TIOBE Index for March 2016 [Електронный ресурс] : Tiobe - The Software Quality Company. — Режим доступу до сайту: http://www.tiobe.com/tiobe_index?page=index
5. Using the Emulator [Електронний ресурс] :Android Developers. — http://developer.android.com/intl/ru/tools/devices/emulator.html#acceleration

Розробка програмного забезпечення для оцінки та зміни параметрів якості растрових зображень

Яворівський Б.

Національний університет «Львівська політехніка», nnelconn@gmail.com

The developed software based on open systems and software is available for operating systems Linux and Windows. It is planned to develop for Android. The application uses assesses the quality bitmaps quantitative indicators. Quality of image is formed on the basis of the calculation and adjustment of brightness, contrast, contrast, tone, contrast, saturation, brightness and tone. Experimental results are shown as histograms. A convolution method to change the color characteristics of the pixels is presented.

Зорова система людини - найбільш надійний і досконалий вимірювальний інструмент для оцінки якості цифрового зображення. Проте суб'єктівна оцінка растрових зображень є досить складною і повільною, для здійснення вимагає залучення досвідчених експертів. Тому розробка програмного забезпечення для оцінки якості та редагування растрових зображень є актуальним завданням. Даний ужиток може бути використане у різних сферах життєдіяльності людини, де необхідно покращувати якість цифрових зображень.

Можливі два підходи до оцінки якості зображень: кількісна оцінка, яка базується на основі використання математичних і суб'єктивна оцінка на основі експертних оцінок [1].

Суб'єктивні та кількісні оцінки якості зображень можуть бути абсолютними або порівняльними. Абсолютна міра якості використовується для оцінки одного зображення, тобто зображенню присвоюється відповідна категорія в рейтинговій шкалі. Порівняльні заходи використовуються для ранжування набору зображень в якісній шкалі від «найкраще» до «найгірше» або взаємного порівняння двох зображень, наприклад, вихідного і відфільтрованого (чи отриманого у різні дні, різними камерами і т.д.).

Міру різкості зображення можна визначити шляхом знаходження кута нахилу профілю яскравості зображення на кордоні перепаду. Для оцінки контрасту зображення здіснюють порівняння пікселів, виходячи з окремих комбінацій елементів зображення. При цьому всі елементи вважаються