МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

О. С. Губка, С. О. Губка

ОСОБЛИВОСТІ ТЕСТУВАННЯ МОБІЛЬНИХ ДОДАТКІВ

Навчальний посібник

Харків "ХАІ" 2020

УДК 004.054
Г 28

Рецензенти: д-р техн. наук, проф. І. П. Гамаюн,
д-р техн. наук, проф. А. Л. Литвинов

Губка, О. С.

Г 28 Особливості тестування мобільних додатків [Текст] : навч. посіб. О. С. Губка, С. О. Губка. – Харків : Нац. аерокосм. ун-т ім. М. Є. Жуковського «Харків. авіац. ін-т », 2020. – 80 с.

ISBN 978-966-662-752-3

Описано загальну різницю між мобільними платформами Android та iOS. Розглянуто загальні етапи, особливості та принципи тестування мобільних додатків. Наведено приклад шаблону чек листа тестування мобільніх додатків.

Для студентів напрямів «Комп'ютерні науки», «Програмна інженерія», «Комп'ютерна інженерія».

Іл. 15. Табл. 5. Бібліогр.: 6 назв

УДК 004.054

ISBN 978-966-662-752-3

ЗМІСТ

ВСТУП

Мобільний додаток – програмне забезпечення, призначене для роботи на смартфонах, планшетах та інших мобільних пристроях. Багато мобільних додатків встановлені на самому пристрої або можуть бути завантажені на нього з онлайн магазинів мобільних додатків, таких, як App Store, Google Play, Windows Phone Store та інших, безкоштовно або за плату.

Спочатку мобільні додатки використовувалися для швидкої перевірки електронної пошти, але високий попит на них привів до розширення їх призначень і в інших областях, таких, як ігри для мобільних телефонів, GPS, спілкування, перегляд відео та користування інтернетом.

Велика кількість мобільних пристроїв продаються з уже встановленим набором мобільних додатків – веб-браузер, поштовий клієнт, календар (наприклад, Google Calendar), додаток для придбання та прослуховування музики тощо. Деякі попередньо встановлені додатки можуть бути вилучені з мобільного пристрою користувачем, за допомогою звичайного процесу видалення, вивільняючи більше місця для зберігання інших (бажаних) додатків.

Додатки, що відразу не встановлені на мобільний пристрій, доступні для завантаження та встановлення через платформи їх поширювання. Такі платформи називаються магазинами додатків. Вони почали з'являтися у 2008 році та зазвичай керуються компанією-власником мобільних операційних систем: Apple App Store, Google Play, Windows Phone Store і BlackBerry App World. Проте існують свого роду незалежні магазини додатків: Cydia, GetJar або F-Droid. Значна кількість мобільних додатків є безкоштовною для встановлення та користування, але й існують і платні. Всі додатки зазвичай завантажуються з платформи одразу на цільовий пристрій, але іноді вони можуть бути завантаженими на ноутбуки чи комп'ютери. Як правило, 20...30 % вартості платних додатків надходить до компанії-дистриб'ютора (наприклад, iTunes), а решта – до виробника. Тому той самий додаток може мати різну вартість залежно від мобільної платформи.

Додатки також можуть бути встановлені власноруч користувачем, такі як, наприклад, запуск пакету програм для Android на пристроях з операційною системою Android.

Спочатку мобільні додатки пропонувалися як інструменти для контролю та оперування загальними потоками інформації, включаючи електронну пошту, календар, контакти, фондовий ринок та інформацію про погоду. Тим не менше попит і наявність інструментів для розробників зумовили швидке розповсюдження додатків і для інших категорій електронних пристроїв, які функціонують за допомогою настільних

прикладних програм. Як і у випадку з іншим програмним забезпеченням, вибухова кількість та різноманіття мобільних додатків привели до виникнення великої кількості пізнавальних ресурсів – з відгуками, рекомендаціями та оглядами, а саме: блоги, журнали та спеціальні служби виявлення додатків в інтернеті. У 2014 році державні регуляційні служби України почали створювати та регулювати мобільні додатки, зокрема ті, які стосуються медичної індустрії. Деякі компанії навіть почали пропонувати додатки як альтернативний метод надання інформації на противагу офіційним веб-сайтам. Так, наприклад, у 2017 році більш ніж 60 % трафіку на сайти українських мікрофінансових компаній було отримано з мобільних пристроїв.

Користування мобільними додатками серед користувачів мобільних пристроїв стає все більш і більш популярним. Згідно з даними дослідження компанії AppAnnie за 2017 рік кількість завантажень додатків зросла на 60 %, споживчі витрати зросли більш ніж удвічі, а час, витрачений на додаток кожним користувачем, становить близько 43 днів на рік. Дослідники виявили, що використання мобільних додатків чітко корелюється з їх наповненням і залежить від часу доби і локації розміщення користувача. Мобільні додатки відіграють все більш значущу роль у галузі охорони здоров'я, а при врахуванні правильного дизайну та інтеграції – можуть мати багато переваг. Також компанія AppAnie прогнозує зростання ринку мобільних додатків на 270 % до 2020 року. Найбільше прибутків будуть приносити мобільні ігри - 55 % від загальної суми.

Компанія з дослідження ринку Gartner передбачила, що протягом 2013 року приблизна кількість завантажених мобільних додатків становитиме 102 мільярди (91 % з яких будуть безкоштовними), що своєю чергою становитиме 26 мільярдів доларів США (а це на 44,4 % більше порівняно з 18 мільярдами доларів США за 2012 рік). Згідно з даними до другого кварталу 2015 року самі лише магазини Google Play і Apple Store зібрали 5 мільярдів доларів.

Розроблення програмного забезпечення для мобільних пристроїв потребує врахування їх обмежень і можливостей. Мобільні пристрої працюють на акумуляторі та мають менш потужні процесори, ніж персональні комп'ютери, а також мають більше функцій, таких, як визначення місцезнаходження та камери. Розробникам також доводиться враховувати широкий спектр розмірів дисплея, різні технічні характеристики та конфігурації обладнання через сильну конкуренцію мобільного програмного забезпечення та зміни в кожній платформі.

Розроблення програм для мобільних платформ потребує використання спеціалізованих інтегрованих середовищ розроблення. Мобільні додатки спочатку тестуються в середовищі розроблення з використанням емуляторів, а потім перевіряються на місцях. Емулятори

забезпечують недорогий спосіб тестування програм на мобільних телефонах, до яких розробники можуть не мати фізичного доступу.

Дизайн мобільного інтерфейсу користувача також має важливе значення. Мобільний інтерфейс розглядає обмеження та контексти, екран, вхід і мобільність як контури для дизайну. Користувач часто фокусується на взаємодії з пристроєм, а інтерфейс використовує компоненти як апаратного, так і програмного забезпечення. Вхід користувача дозволяє користувачам керувати системою, а випуск пристрою дозволяє системі вказувати ефекти управління користувачами. Особливості дизайну мобільного інтерфейсу містять: обмеження з боку інтерфейса користувача та форм-фактори – розмір екрана мобільного пристрою для користувача. Контексти мобільного інтерфейсу сигналізують про активність користувача – розташування та планування, які можуть відображуватися внаслідок взаємодії користувачів у мобільній програмі. У цілому мета дизайну мобільного інтерфейсу – це, в першу чергу, зрозумілий, зручний інтерфейс.

Мобільні інтерфейсні системи (front-end) покладаються на мобільні резервні копії (back-end) для підтримки доступу до корпоративних систем. Мобільний зворотний зв'язок полегшує маршрутизацію даних, безпеку, аутентифікацію, авторизацію, роботу поза мережею. Ця функціональність підтримується різноманіттям компонентів проміжних програм, включаючи сервери мобільних додатків, Mobile Backend як службу (MBaaS) та інфраструктуру SOA.

Розмовні інтерфейси відображують комп'ютерний інтерфейс і взаємодіють через текст, а не графічні елементи. Вони імітують розмови зі звичайними людьми. Є два основних типи розмовних інтерфейсів: голосові помічники (як Amazon Echo) і chatbots.

1. ОСОБЛИВОСТІ АРХІТЕКТУРИ ТА РЕАЛІЗАЦІЇ МОБІЛЬНИХ ДОДАТКІВ

На сьогодні ринок мобільних додатків є таким, що найбільше розвивається. До цього приводять здешевлення і одночасно збільшення потужності мобільних пристроїв, розвиток бездротових засобів зв'язку і зміна загальної структури інформаційної взаємодії. Мобільні пристрої набувають все більшого розповсюдження в різних областях діяльності. Зараз вони використовуються не тільки в розважальних цілях, але і в ділових: для різноманітного спілкування (відео, аудіо, текстового), отримання інформації за допомогою електронної пошти та різних розсилок, як сховище даних (довідники, словники), для підготовки документів. По суті сучасні мобільні пристрої часто замінюють пристрої мобільного офісу (ноутбуки та компютери).

Тестування мобільних додатків набуває ще більшої значущості в зв'язку з високою конкуренцією додатків. Користувач очікує надійної, зрозумілої і функціональної програми. Якщо його очікування не виправдовуються, він легко переключається на конкуруючий продукт. Мобільні програми досить дешеві, часто мають безкоштовні версії і дуже легко встановлюються, тому кінцевий користувач не прив'язується до бренду так сильно, як в десктопних додатках.

На розроблення мобільних додатків накладають відбиток особливості архітектури пристроїв і мобільних операційних систем, апаратні обмеження, спосіб взаємодії з пристроями та інші фактори.

Специфіка додатків відбивається і на самому процесі розроблення. Більшість проектів для мобільних пристроїв є короткостроковими і виконуються невеликими командами. Таке розроблення легко вкладається в рамки гнучких методологій. Утім незалежно від моделі розроблення швидше за все не буде формалізованих і детальних специфікацій і тестової документації. Зазвичай все обмежується набором вимог і дизайном користувальницького інтерфейсу. При цьому існує велика мінливість як для користувача вимог і функціональності програми (дуже динамічний ринок), так і апаратно-системних обмежень (з'являються нові пристрої і версії операційної системи).

1.1. Особливості версій мобільної операційної системи iOS

iOS – це власницька мобільна операційна система від Apple. Розроблена спочатку для iPhone, згодом також удосконалена для використання на iPad (до літа 2019, коли на конференції Apple WWDC було представлено нову OS для iPad – iPadOS), iPod Touch та Apple TV (до 9 вересня 2015, коли на спеціальному заході Apple було представлено

tvOS). Apple не дозволяє роботу ОС на мобільних телефонах інших фірм. Відома як iPhone OS до червня 2010 року зараз iOS є похідною від OS X, отже, є за своєю природою Unix-подібною операційною системою.

Інтерфейс користувача iOS основано на концепції прямої маніпуляції з використанням жестів Multi-Touch. Елементи інтерфейсу управління складаються з повзунків, перемикачів і кнопок. Він призначений для безпосереднього контакту користувача з екраном пристрою. Внутрішній акселерометр використовуються деякими програмами для реагування на, переміщення пристрою і являє собою стандартну команду або скасування або обертання пристрою у трьох вимірах, або перемикання між книжковим та альбомним режимами інтерфейсу.

Станом на 2019 рік інтернет-магазин App Store налічує понад 2 мільйони додатків для iOS, які були завантажені понад 15 мільярдів разів. Станом на травень 2010 року iOS становила 15,4 % ринку операційних систем для смартфонів - третя після Symbian і Blackberry.

Як операційна система iOS була представлена з iPhone на Macworld Conference & Expo 9 січня 2007 року і випущена в червні того ж року. Спершу Apple не вказувала її ім'я, просто заявивши, що "iPhone використовує OS X". Спочатку сторонні програми не підтримувалися. Стів Джобс заявив, що розробники можуть створювати веб-програми, які „будуть поводити себе як рідні програми на iPhone“. 17 жовтня 2007 року Apple оголосила, що рідний SDK знаходиться в стадії розроблення і що компанія планує віддати його „в руки розробників у лютому“. 6 березня 2008 року Apple випустила першу бета-версію, а також одержала нове ім'я для операційної системи: iPhone OS. Продажі мобільних пристроїв Apple викликали інтерес до SDK. Apple також продала понад одного мільйона iPhones під час курортного сезону 2007. У червні 2010 року Apple перейменувала iPhone OS на iOS. Назва iOS використувалася компанією Cisco вже більше десяти років на маршрутизаторах Cisco. Для того щоб уникнути будь-якого потенційного позову, Apple ліцензувала торгову марку iOS у Cisco.

Хронологія версій iOS:

- **1.0** – червень 2007 року, перша версія.
- **2.0** – 11 липня 2008 року, підтримка iPhone SDK і App Store, 3G, GPS.
- **3.0** – 17 червня 2009 100 нових функцій, включаючи: вирізати/ копіювати/вставити, MMS, Spotlight, Speak Notes, можливість скачувати TV-шоу, музичні відео, фільми та аудіокниги прямо на iPhone, Find My iPhone, пересилання/видалення SMS та багато іншого.
- **4.0** – 21 червня 2010 (анонсована 7 червня 2010). Понад 100 нових функцій, включаючи багатозадачність, і понад 1500 нових API для розробників додатків. Сумісна з iPhone 3G, iPhone 3G S, iPhone 4, iPod

Touch другого, третього і четвертого поколінь.

- **5.0** – 12 жовтня 2011. Нова версія операційної системи була анонсована 6 червня 2011 під час конференції WWDC. Оголошено про більш ніж 200 нових функцій, включаючи зміни в режимі оповіщень, можливість оновлення ПЗ без використання комп'ютера, програму iMessage, «натуральну» інтеграцію з Twitter і понад 1500 нових API для розробників додатків. Версія сумісна з iPhone 3GS, iPhone 4, iPhone 4S, iPod Touch третього і четвертого поколінь, а також iPad усіх поколінь.
- **6.0** – 19 вересня 2012.
- **7.0** – 18 вересня 2013. Фінальний реліз. Має новий сучасний інтерфейс і багато корисних функцій.
- **8.0** – 17 вересня 2014 року. Поліпшена стабільність. Містить додаток Health, який дозволяє слідкувати за здоров'ям. Інтегрований сервіс з пошуку музики Shazam у Siri.
- **9.0** - 16 вересня 2015 року. Поліпшена стабільність. Збільшені продуктивність і безпека. Siri стала розумнішою. Тепер вона пропонує варіанти ще до того, як Ви поставили запитання.
- **10.0** - версія операційної системи iOS від корпорації Apple, наступник iOS 9 і попередник iOS 11. Була представлена на конференції Worldwide Developers Conference 13 червня 2016 року. Фінальна версія стала доступна 13 вересня 2016 року.
- **11.0** - версія операційної системи iOS від корпорації Apple, наступник iOS 10 і попередник iOS 12. Була представлена на конференції Worldwide Developers Conference 5 червня 2017 року. Перша бета-версія стала доступна для розробників відразу після основної презентації, публічне тестування почалося в липні 2017 року. Фінальний реліз iOS 11 для споживачів відбувся 19 вересня 2017 року.
- **12.0** - 17 вересня 2018 року корпорація Apple випустила першу релізну версію операційної системи Apple iOS 12.0.

Особливості:

- Відеовиклик FaceTime на 32 людини (для старих пристроїв недоступні).
- Memoji можна використовувати в повідомленнях і FaceTime (Для iPhone X, iPhone Xs, iPhone Xs Max, iPhone Xr).
- Функція «Екранний час» показує, скільки часу Ви проводите за пристроєм.
- Швидке налаштування повідомлень, угруповання повідомлень.
- Оновлений режим «Не турбувати».
- Оновлений «Загальний доступ до фото» і новий розділ «Для вас» у додатку «Фото» для iOS.
- Швидкі команди Siri.
- Новий додаток Apple Books.

- Новий дизайн програми «Акції».
- Диктофон на iPad (не для всіх пристроїв).
- Нові пропоновані паролі.
- Заповнити форму коду з Повідомлень.
- Сторонні додатки для навігації в CarPlay - тільки в деяких країнах (США, країни Європи).
- Підвищено продуктивність на всіх підтримуваних пристроях.

13.0 - тринадцята версія мобільної операційної системи iOS, розробленої Apple, є наступницею iOS 12. Новий великий реліз був анонсований на Apple Worldwide Developers Conference 3 червня 2019 року і випущений 19 вересня 2019 року.

iOS 13 і iPadOS були представлені на презентації для розробників (WWDC19) 3 червня 2019 року. Перша бета-версія доступна тільки для зареєстрованих розробників. Очікується, що публічна бета-версія буде доступна в липні 2019 року, а остаточний реліз для всіх користувачів - восени 2019 року.

Особливості:

- Dark Mode (темний режим) - був представлений «темний режим», що дозволяє включати темну колірну схему для системних додатків, а також для призначеного для користувача інтерфейсу iOS і iPadOS. Компанія Apple заявляє, що розробники сторонніх додатків з AppStore також можуть додавати «темний режим» до своїх додатків.
- Оновлення клавіатури - вбудована клавіатура iOS 13 тепер має QuickPath - функцію, що дозволяє користувачеві проводити пальцем по клавіатурі, щоб скласти речення або ввести будь-яке слово. Раніше ця функція була доступна виключно через деякі сторонні додатки для клавіатури з AppStore.

1.2. Особливості мобільної операційної системи Android

Android – операційна система і платформа для мобільних телефонів і планшетних комп'ютерів, створена компанією Google на базі ядра Linux. Підтримується альянсом Open Handset Alliance (OHA).

Хоча Android базується на ядрі Linux, він стоїть дещо осторонь Linux-спільноти та Linux-інфраструктури. Базовим елементом цієї операційної системи є реалізація Dalvik віртуальної машини Java, і все програмне забезпечення і застосування спираються на цю реалізацію Java.

У 84 % смартфонів, проданих у III-му кварталі 2014 року, була встановлена операційна система Android.

У березні 2017 року ОС Android стала найпопулярнішою ОС, з якої виходили в інтернет. Так, 37,93 % користувачів заходили в інтернет із Android, а з Windows – 37,91 % користувачів. В Азії показники ще вищі –

52,2 і 29,2 % відповідно.

Android, Inc. було засновано в Пало-Альто (Каліфорнії, США), у жовтні 2003 року Енді Рубіном (співзасновник компанії Danger), Річардом Майнером (співзасновник Wildfire Communications, Inc.), Ніком Сірсом (колишній віце-президент компанії T-Mobile) і Крісом Уайтом (очолював дизайн і розроблення інтерфейсу в WebTV) для розроблення. За словами Рубіна, «більш розумних мобільних пристроїв, які краще знають про місце перебування власника і його вподобання». Попередні наміри компанії полягали в тому, щоб розробити вдосконалену операційну систему для цифрових фотоапаратів, але було зрозуміло, що ринок пристроїв не був достатньо великим, і вони спрямували свої зусилля на розроблення операційної системи для смартфонів, щоб конкурувати з Symbian і Windows Mobile (на той момент Apple ще не випустила iPhone). Незважаючи на минулі досягнення засновників і колишніх співробітників, Android, Inc. працювала таємно, показуючи тільки те, що вона розробляла програмне забезпечення для мобільних телефонів. Цього ж року Рубін залишився без грошей. Однак Стів Перлман (близький друг Рубіна) приніс йому 10 тис. доларів готівкою у конверті та відмовився від своєї частки в компанії.

У липні 2005 року компанія Google купила Android, Inc. Усі засновники цієї стартап-компанії пішли працювати у Google. На той час мало що було відомо про Android, Inc. окрім того, що вони займаються розробленням ПЗ для мобільних телефонів. Такий розвиток подій спричинив виникнення чуток про те, що Google планує увійти на ринок мобільних телефонів, але було незрозуміло, що саме компанія планує там робити.

У Google група на чолі з Рубіном розробила ОС на основі Linux (ядро v.2.6), яку вони пропонували розробникам телефонів та операторам мобільного зв'язку як гнучку та швидку систему. Повідомлялося, що Google планує співпрацю з рядом розробників апаратних компонентів і програмного забезпечення, і вона є відкритою для співпраці з операторами мобільного зв'язку.

У грудні 2006 року знову пішли чутки про те, що Google буде просуватися на ринок мобільних телефонів. У доповіді BBC і The Wall Street Journal зазначалося, що Google хоче розмістити пошуковик Google і ПЗ Google на мобільних телефонах, і компанія постійно напружено працює для досягнення цієї мети.

Далі у пресі та онлайн-ЗМІ почали з'являтися чутки, що Google розробляє телефон під власним брендом, а були й інші, які стверджували, що Google визначила технічні характеристики та вже презентує прототипи розробникам телефонів та операторам мобільного зв'язку. Повідомлялося, що буде реалізовано приблизно 30 прототипів. Network World повідомляє, що Google-телефон дійсно є телефоном з відкритою операційною системою на відміну від схожих продуктів, таких, як iPhone. Цей проект створення смартфона з використанням відкритого коду базувався в тому

числі на ядрі Linux.

5 листопада 2007 року консорціум Open Handset Alliance (OHA) заявив про намір розробити відкриті стандарти для мобільних пристроїв. У той же день концерн подав як свій перший продукт платформу для мобільних телефонів на основі ОС Linux – Android.

Перша версія Android була випущена 23 вересня 2008 року і називалася 1.0 Astroboy, а наступна – 1.1 Bender. Від назв на честь відомих роботів згодом довелося відмовитися через розбіжності з правовласниками. Від 2008 року Android пережив численні оновлення, які поступово покращували операційну систему, додаючи нові функції, виправляли помилки у попередніх випусках. Тепер кодове ім'я кожного великого релізу Android, починаючи з версії 1.5, є назвою будь-якого десерту. Перші букви найменувань у порядку версій відповідають літерами латинського алфавіту:

- 1.5 Cupcake («кекс»).
- 1.6 Donut («пончик»).
- 2.0/2.1 Eclair («еклер» або «глазур»).
- 2.2 Froyo (скорочення від «заморожений йогурт»).
- 2.3 Gingerbread («імбирний пряник»).
- 3.0 Honeycomb («медові стільники»).
- 4.0 Ice Cream Sandwich («брикет морозива»).
- 4.1/4.2/4.3 Jelly Bean («желейні боби»).
- 4.4 KitKat (на честь однойменного бренду шоколадних батончиків; раніше планувалася назва Key Lime Pie).
- 5.0/5.1 Lollipop («льодяник»).
- 6.0/6.1 Marshmallow («зефір»).
- 7.0/7.1 Nougat – Nougat («нуга»).
- 8.0/8.1 Oreo – Oreo (печиво «Орео»).
- 9.0 Pie – Pie («пиріг»).

1.3. Статистична інформація щодо мобільних пристроїв в Україні на жовтень 2019 року

Статистику щодо виробників мобільних пристроїв показано на рис. 1.1, а статистику за версиями ОС Android - на рис. 1.2, статистику за версіями iOS - на рис. 1.3, статистику щодо роздільних здатностей екранів мобільних пристроїв - на рис.1.4, статистику найбільш популярних мобільних пристроїв - на рис. 1.5.

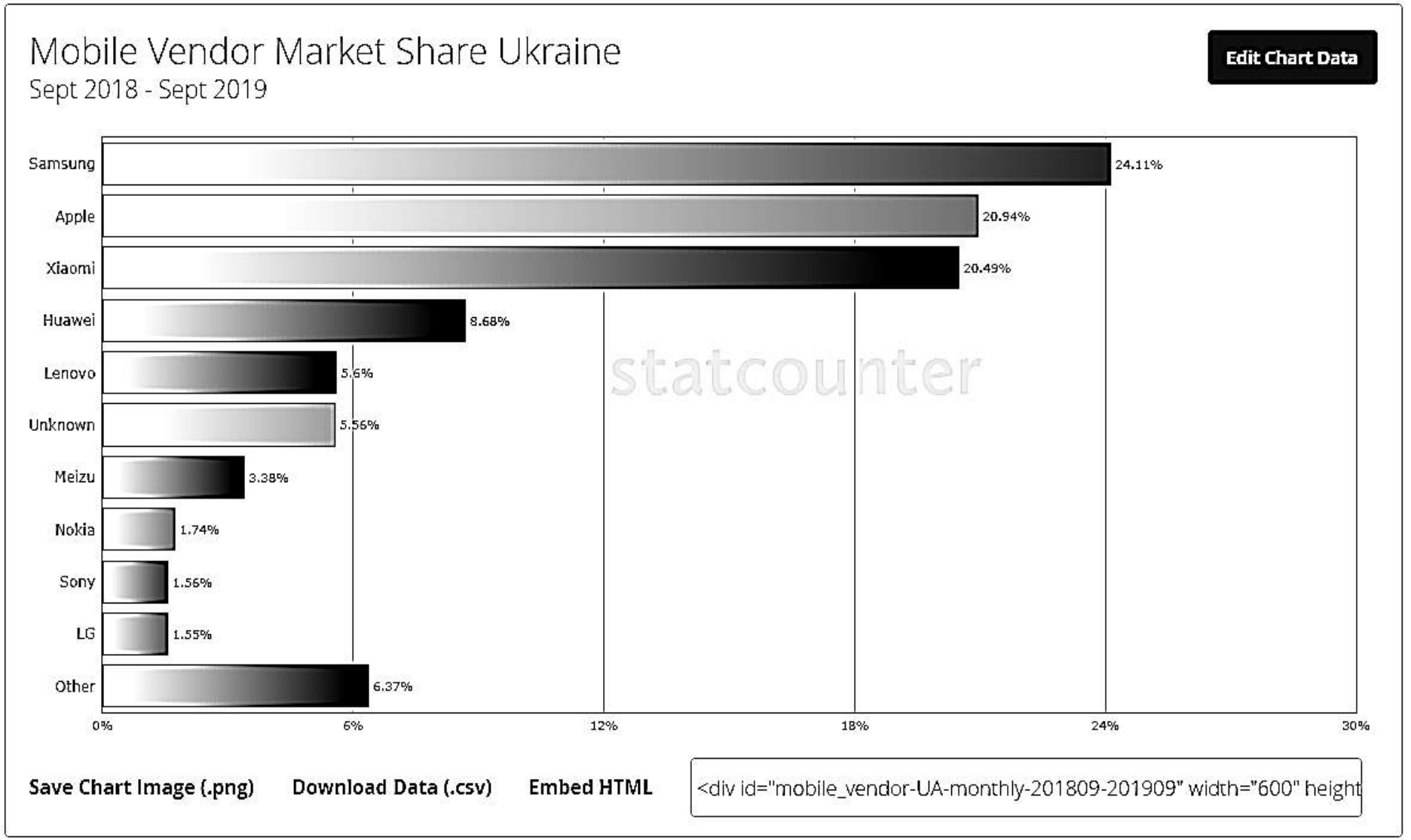


Рис. 1.1. Статистика щодо виробників мобільних пристроїв

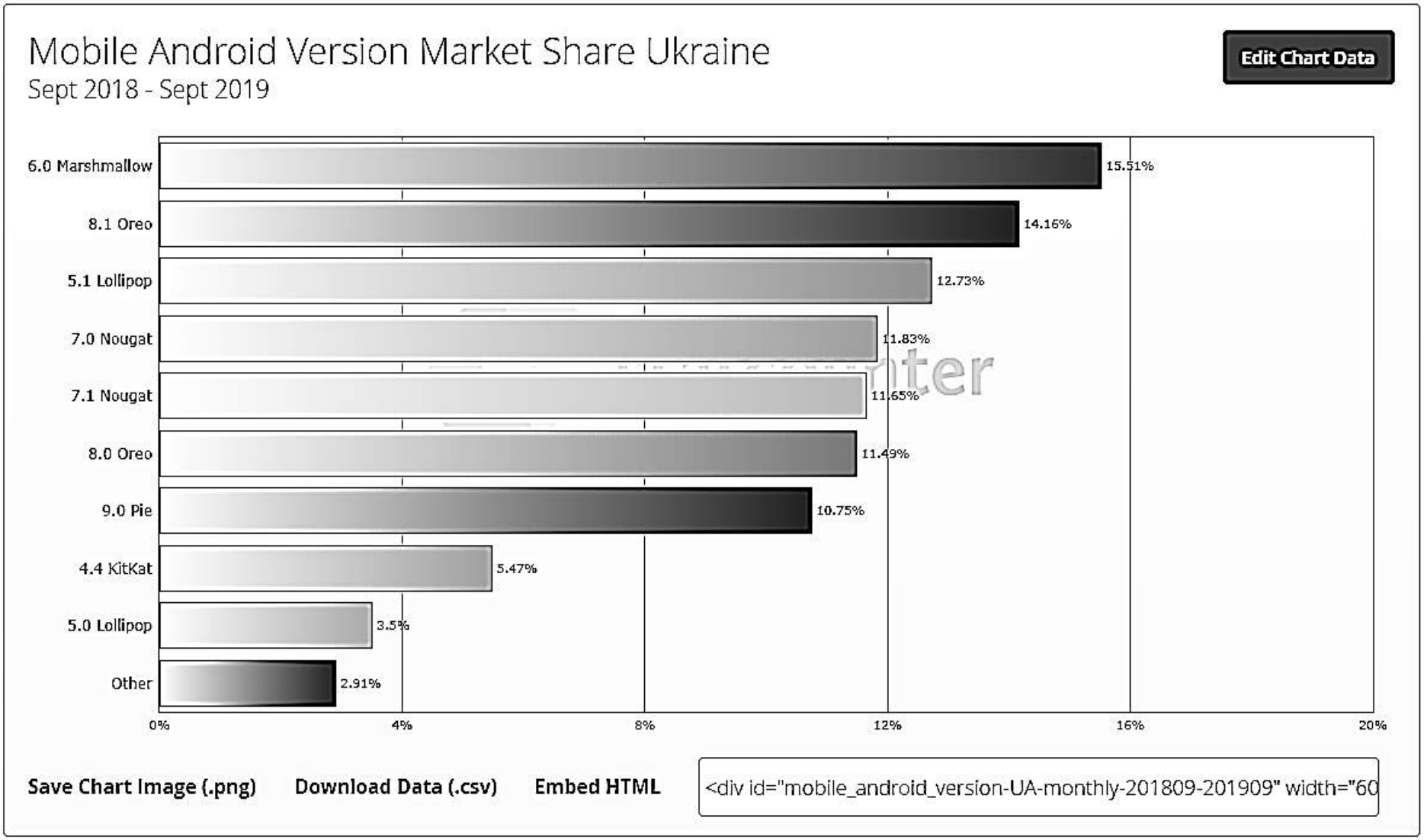


Рис. 1.2. Статистика за версіями операційної системи Android

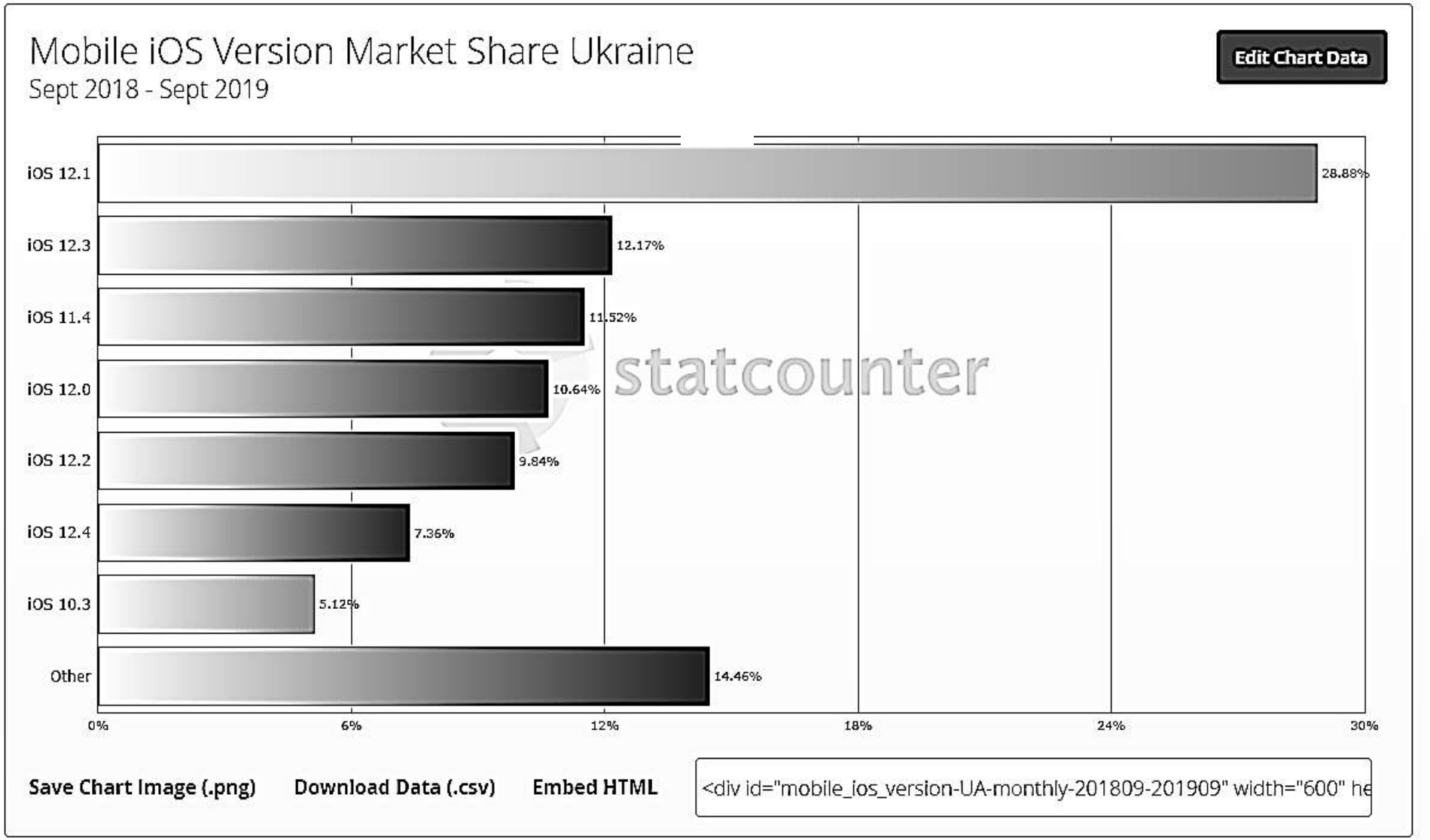


Рис. 1.3. Статистика за версіями операційної системи iOS

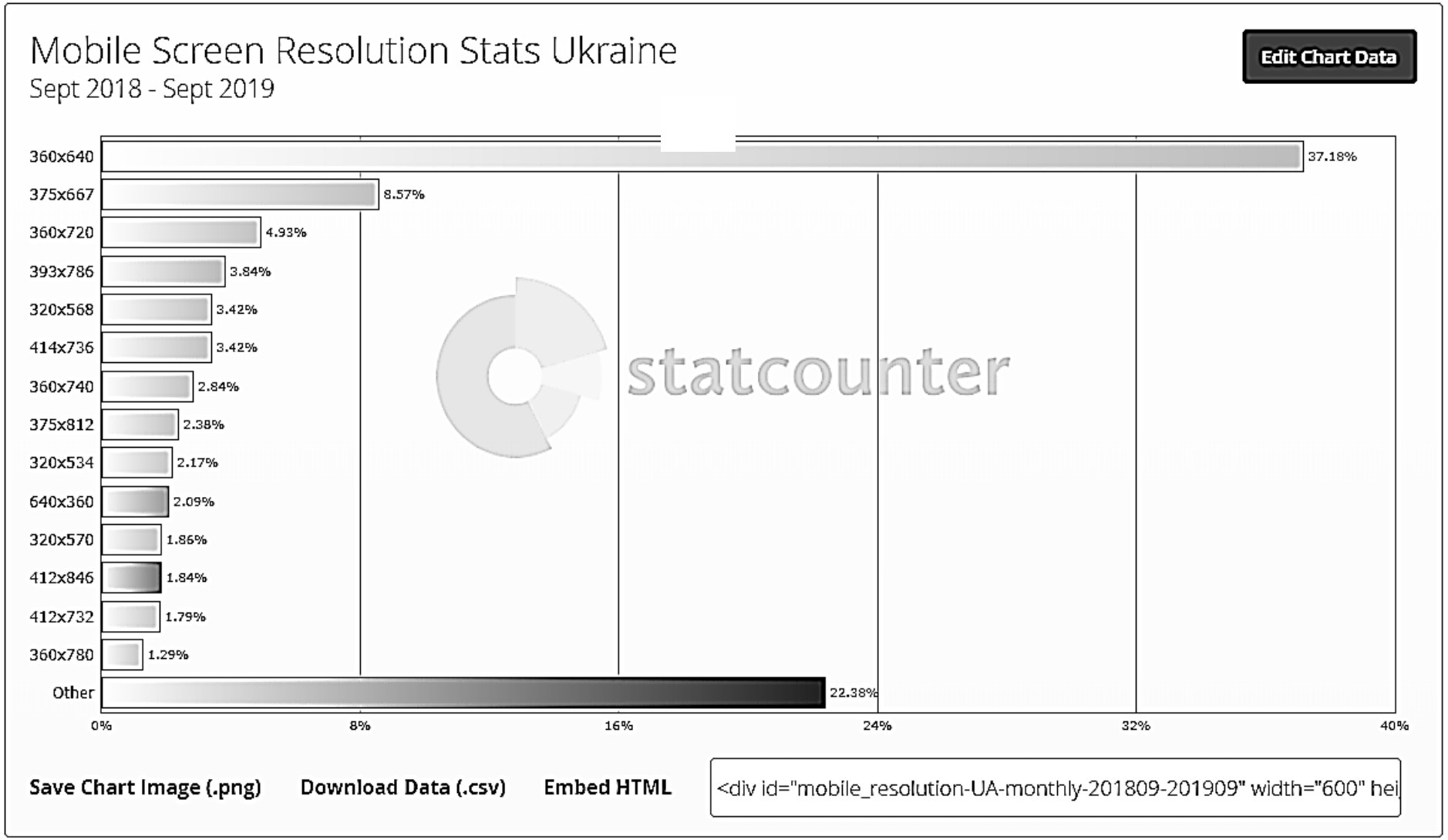
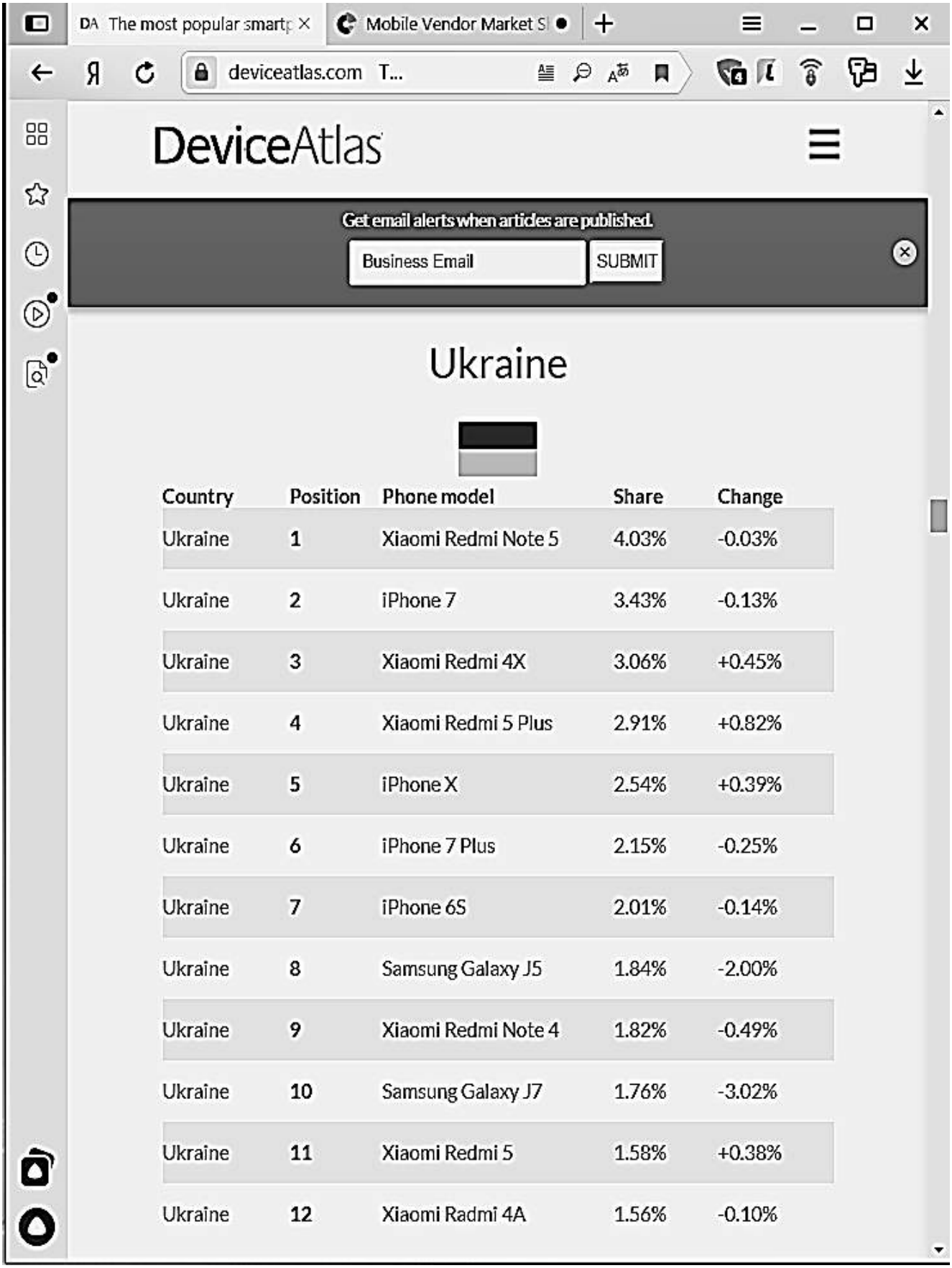


Рис. 1.4. Статистика щодо роздільних здатностей екранів мобільних пристроїв



Country	Position	Phone model	Share	Change
Ukraine	1	Xiaomi Redmi Note 5	4.03%	-0.03%
Ukraine	2	iPhone 7	3.43%	-0.13%
Ukraine	3	Xiaomi Redmi 4X	3.06%	+0.45%
Ukraine	4	Xiaomi Redmi 5 Plus	2.91%	+0.82%
Ukraine	5	iPhone X	2.54%	+0.39%
Ukraine	6	iPhone 7 Plus	2.15%	-0.25%
Ukraine	7	iPhone 6S	2.01%	-0.14%
Ukraine	8	Samsung Galaxy J5	1.84%	-2.00%
Ukraine	9	Xiaomi Redmi Note 4	1.82%	-0.49%
Ukraine	10	Samsung Galaxy J7	1.76%	-3.02%
Ukraine	11	Xiaomi Redmi 5	1.58%	+0.38%
Ukraine	12	Xiaomi Radmi 4A	1.56%	-0.10%

Рис. 1.5. Статистика найбільш популярних мобільних простоїв

1.4. Типи мобільних додатків

З ростом кількості мобільних пристроїв збільшується і потреба в розробленні додатків. Щодня створюються сотні мобільних додатків, які виконують функції з мобільним пристроєм більш комфортною. Існує кілька типів додатків, які використовують розробники.

Нативні додатки

Ці додатки називають нативними тому, що вони написані рідною (з англ. Native - рідна) для певної платформи мовою програмування. Для Android цією мовою є Java, тоді як для iOS - objective-C або Swift.

Нативні додатки знаходяться на самому пристрої, доступ до яких можна отримати, натиснувши на іконку. Вони встановлюються через магазин додатків (Play Market на Android, App Store на iOS та ін.).

Розроблені спеціально для конкретної платформи і можуть використовувати всі можливості пристрою - камеру, GPS-датчик, акселерометр, компас, список контактів тощо. Також вони можуть розпізнавати стандартні жести, встановлені операційною системою, або зовсім нові жести, які використовуються в конкретному додатку.

Плюси нативних додатків:

- швидкість роботи і продуктивність;
- високий ступінь безпеки;
- розширений інтерфейс;
- відносно висока вартість розроблення;
- максимально можлива функціональність;
- здатність працювати без Інтернету;
- зручність для кінцевого користувача.

Мінуси нативних додатків:

- охоплення платформ;
- тривалі терміни розроблення;
- необхідність випускати оновлення в косметичних цілях.

Через те що нативні додатки оптимізовані під конкретну ОС, вони органічно вписуються в будь-який смартфон, відрізняючись високою швидкістю роботи і продуктивністю.

Нативні додатки можуть отримати доступ до системи оповіщення пристрою, а також залежно від призначення нативного додатку вони повністю або частково обходиться без наявності інтернет-з'єднання.

Приклади нативних додатків:

Перший приклад - додаток Shazam, який здійснює визначення та пошук інформації, програє на іншому пристрої пісні:

- встановлюєм з магазину додатків;
- для роботи необхідний доступ до інтернету;
- використовує диктофон телефону.

Другий приклад - додаток Instagram:

- встановлюєм з магазину додатків;
- для роботи також необхідний доступ до інтернету;
- використовує ПЗ смартфона: камеру, геолокацію, адресну книгу;
- можна налаштувати отримання push-повідомлень.

Мобільні веб-додатки

Насправді мобільні веб-додатки не є додатками як такими. Адже справа в тому, що веб-додаток по суті являє сайт, який адаптований і оптимізований під будь-який смартфон, і для того щоб скористатися ним, достатньо мати на пристрої браузер, знати його адресу і розташовувати інтернет-з'єднанням (завдяки йому відбувається оновлення інформації в цьому виді додатків).

Запускаючи мобільні веб-додатки, користувач виконує всі ті дії, які він виконує при переході на будь-який веб-сайт, а також отримує можливість «встановити» їх на свій робочий стіл, створивши закладку сторінки веб-сайту.

Веб-додатки відрізняються кросплатформеністю, тобто вони здатні функціонувати незалежно від платформи девайса. Перевагою цих додатків є те, що вони не використовують його програмне забезпечення, а через те що вони є мобільною версією сайту з розширеним інтерактивом, веб-додатки не займають дорогоцінне місце в пам'яті смартфона.

Веб-додатки стали широко популярні в той час, коли почав розвиватися HTML5 і люди усвідомили, що можуть отримати доступ до безлічі функцій нативних додатків, просто зайшовши на веб-сайт через звичайний браузер. На сьогодні складно сказати, де саме розташовується чітка межа між веб-додатками і звичайними веб-сторінками, оскільки функціонал HTML5 зростає з кожним днем і все більше і більше сайтів його використовують.

Розробляються веб-додатки за допомогою інструментів і фреймворків, які стали традиційними. Внаслідок цього процес їх розроблення останнім часом істотно прискорився. Фахівців з їх розроблення достатньо.

У той же час до недоліків веб-додатків слід віднести нездатність працювати з ними без інтернету. Причому з цього випливає й інший мінус - їх продуктивність, яка знаходиться на середньому рівні порівняно з іншими

видами додатків. Більш того вона залежить від можливостей інтернет-з'єднання провайдера послуг.

Плюси мобільних веб-додатків:

- повне охоплення платформ;
- простий і швидкий процес розроблення;
- кількість компетентних розробників;
- відсутність необхідності завантаження з магазину додатків.

Мінуси мобільних веб-додатків:

- обов'язкове підключення до Інтернету;
- примітивний інтерфейс програми;
- неможливість відправити push-повідомлення;
- продуктивність і швидкість роботи;
- незадовільний рівень безпеки.

Приклади мобільних веб-додатків:

1. last.fm уважається веб-додатком, хоча по суті - це в той же час і веб-сайт.
2. maps.google.com - веб-сайт, але в той же час - це і веб-додаток.

Гібридні додатки

Гібридні додатки являють собою поєднання веб і нативних додатків: маються на увазі їхні кросплатформеність і доступ до функціоналу смартфона. Такі додатки можуть бути завантажені виключно з маркетів на кшталт Google Play і App Store. Разом з тим вони мають у своєму розпорядженні опцію автономного оновлення інформації, а для їх роботи необхідно інтернет-підключення. Без наявності останнього веб-функції просто не працюють.

У більшості компаній вибір найчастіше падає на розроблення саме гібридного додатка. Це можна пояснити тим, що гібридні додатки здатні поєднувати переваги нативних додатків (технологічною актуальністю, яка забезпечується останніми веб-технологіями). Однак на відміну від нативних додатків вартість створення гібридних на порядок нижче, а його швидкість - вище. Спорідненість гібридних додатків з веб-додатками своєю чергою дає плоди у вигляді того, що в них можна легко і оперативно вносити корективи. Тобто розробникам не доводиться, як у випадку з нативними, повторно розміщувати додаток в магазині заради усунення помилок попередньої версії.

Розроблення гібридного додатка є перспективним ще й тому, що він створюється відразу під дві платформи. Як наслідок це позбавляє проблем, пов'язаних з окремим розробленням додатка під кожну ОС. Чи є це вирішальним фактором? Безумовно.

Крім усього іншого, потрібно взяти на замітку, що якість і можливості гібридних додатків залежать, перш за все, від фреймворка, який використовує розробник. Також варто приділити належну увагу факторам, які роблять гібридні додатки гарним варіантом на тлі інших.

Отже, варто розробляти їх, якщо:

- є необхідність заощадити бюджет;
- потрібно створити відносно нескладний додаток з простою анімацією;
- є завдання оперативного розроблення програми як мінімум на дві платформи.

Плюси гібридних додатків:

- вартість і швидкість розроблення;
- кількість розробників;
- кросплатформеність;
- опція автономного поновлення.

Мінуси гібридних додатків:

- некоректна робота при відсутності інтернет-з'єднання;
- середня швидкість роботи на тлі нативних;
- мінімалізм щодо візуальних елементів.

Приклади гібридних додатків:

Перший приклад - додаток HeartCamera для iOS, що дозволяє прикрасити фотографію мальованими серцями і т. п.:

- завантажується з магазину;
- використовує камеру телефону;
- необхідне підключення до інтернету при бажанні поділитися результатом своєї роботи;
- можна налаштувати push-повідомлення.

Другий приклад - додаток TripCase - органайзер для планування подорожей:

- завантажується з магазину;
- може використовувати геолокацію;
- необхідне підключення до інтернету;
- може використовувати стільникову мережу;
- можна налаштувати push-повідомлення.

У сучасному світі важко уявити собі мобільний пристрій, на якому б не було встановлено жодного додатка.

Мобільний додаток являє собою розроблену програму для планшетів і смартфонів, яка встановлюється на ту чи іншу платформу і має певний функціонал. Простіше кажучи, виконує певні дії і вирішує поставлене коло питань.

Вибір відповідної моделі мобільного додатка - це дуже важливий етап в його розробленні, на який впливають кілька факторів, таких, як технічна оцінка розробників, потреба в доступі до інформації на пристрої, вплив швидкості інтернету на додаток тощо.

2. ЕТАПИ ТЕСТУВАННЯ МОБІЛЬНИХ ДОДАТКІВ

Забезпечення якості є невід'ємною частиною життєвого циклу розроблення усіх додатків, включаючи мобільні. На жаль, багато розробників не беруть до уваги критичні особливості тестування мобільних додатків, які часто призводять до збоїв, помилок у роботі програми і поганої якості обслуговування клієнтів.

Аби забезпечити успішне розроблення будь-якої програми, фахівець-тестувальник повинен брати участь в усіх етапах розроблення - від створення концепції і аналізу вимог до створення специфікацій тестування і випуску готового продукту. Забезпечення якості також є ключовим елементом у наступних після проходження етапах розробки, версіях програмного продукту.

Однак часто буває складно визначити, з чого почати організацію процесу тестування мобільного додатка. Для безпроблемного тестування ми рекомендуємо просто виконати дев'ять зазначених нижче кроків.

Розглянемо особливості тестування мобільних додатків. Цикл життя спринтів показано на рис. 2.1.

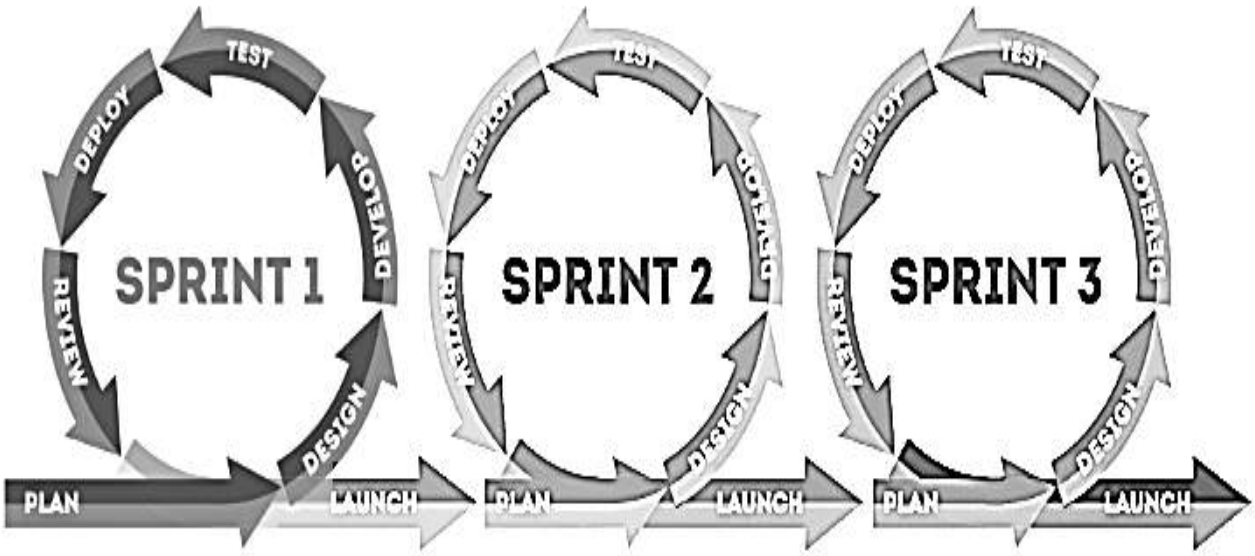


Рис. 2.1. Цикл життя спринтів

Етап 1. Планування

Коли етап розроблення додатка майже завершено, Ви повинні знову поставити перед собою питання - чого Ви намагаєтеся досягти розробленням цього додатка і які є обмеження.

Необхідно визначити таке:

1. Чи взаємодіє ваш додаток з іншими додатками?
2. Наскільки є функціональними всі можливості програми?
3. Чи є тестований мобільний додаток нативним, Mobile-web або гібридним?
4. Чи обмежені завдання тестування програми тестуванням тільки зовнішнього інтерфейсу?
5. Чи варто тестувати backend?
6. Яка повинна бути сумісність з різними бездротовими мережами?
7. Як сильно ці додатки і вільний простір, займаний ним, залежать від особливостей використання програми?
8. Наскільки швидко завантажується ваш додаток, наскільки швидко відбувається серфінг за меню програми та її функцій?
9. Як буде оброблятися можливе збільшення навантаження на додаток?
10. Чи впливають різні зміни в статусі програм телефону на роботу мобільного додатка?

Переконайтеся, чи домовилися Ви з командою тестувальників про роль кожного з них і про ваші очікування від процесу тестування. Зрештою спілкування є ключем до підтримання правильного робочого середовища в команді.

Правильне розуміння ролей і завдань також стосується моменту прописування списку тестових випадків. Уся команда тестувальників (QA) повинна підтримувати і оновлювати цей документ зі звітами з тестування усіх функцій, реалізованих протягом усього процесу розроблення.

Етап 2. Визначення необхідних типів тестування мобільних додатків

Перед тестуванням будь-яких мобільних додатків визначте, що саме в цьому мобільному додатку Ви бажаєте протестувати: набір функціональності, зручність використання, сумісність, продуктивність, безпеку і т. д. На цьому ж етапі має сенс вибрати методи тестування мобільного додатка.

Визначте, на які цільові пристрої спрямований цей додаток і які вимоги до функціоналу слід перевірити.

Ви також повинні визначити, які цільові пристрої потрібно включити до списку тестування.

Ви можете зробити це в такий спосіб:

- з'ясувати, які пристрої буде підтримувати додаток;
- визначити, яка версія операційної системи буде найбільш ранньою з тих, що підтримуються додатком;

• виявити найбільш популярні моделі мобільних пристроїв у цільовій аудиторії;

• визначити набір неосновних (додаткових) пристроїв з екранами різних розмірів, потенційно підтримуваних додатком;

• вирішити, чи будете Ви використовувати для тестування фізичні пристрої або їх емулятори.

Етап 3. Тестові випадки і розроблення сценаріїв тестування програми

Підготуйте документ, що описує тестові випадки (test cases) для кожної тестованої функції і функціональності.

На додаток - до функціональних тестових випадків також повинні бути додані деякі окремі моменти (кейси):

- особливість використання батареї;
- швидкість роботи програми;
- вимоги до даних;
- місткість використовуваної пам'яті.

Також перед початком тестування важливо визначитися, яке поєднання ручного і автоматичного тестувань Ви будете застосовувати.

За необхідності підготуйте окремі набори ручних тестових випадків і сценаріїв для автоматичного тестування і адаптуйте їх відповідно до вимог проекту.

Етап 4. Ручне і автоматичне тестування

Зараз будемо виконувати ручні і автоматизовані тести. На попередніх етапах Ви вже визначили, які тести і скрипти використовувати і підготували їх. На поточному етапі необхідно запустити тести для перевірки механізмів основної функціональності, щоб переконатися у відсутності поломок.

Автоматизоване тестування мобільних додатків економить час та інші ресурси тестувальників.

Етап 5. Тестування юзабіліті і бета-тестування

Після того як базовий функціонал протестовано, слід переконатися, що мобільний додаток є досить простим у використанні і забезпечує задовільний рівень досвіду для користувача. На цьому етапі необхідно підтримувати відповідність матриці платформ, щоб забезпечити охоплення користувачів різних платформ досягнутими бета-тестувальниками (рис. 2.2).

Щойно додаток буде протестовано всередині компанії, Ви зможете випустити бета-версію програми на ринок.

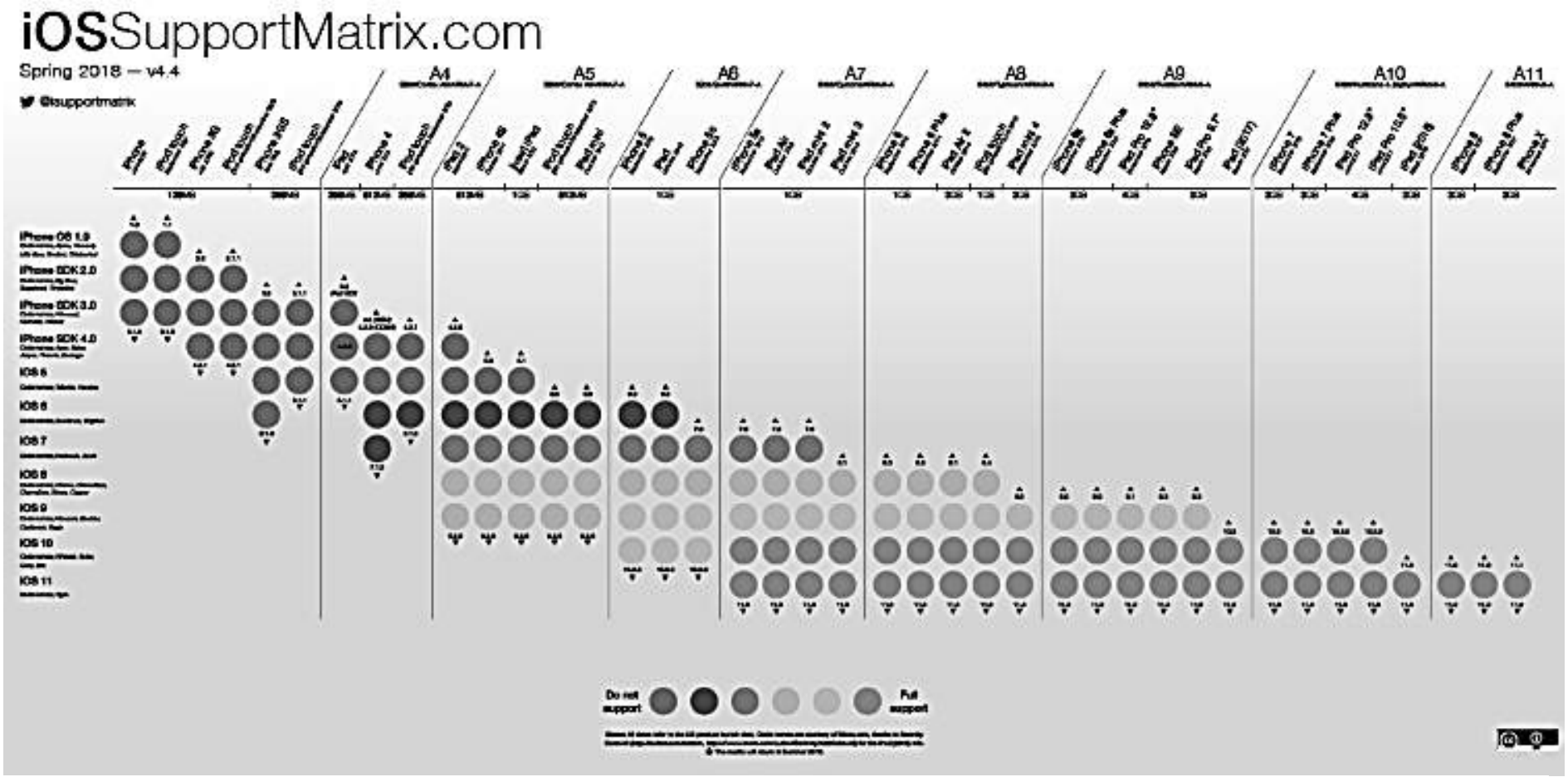


Рис. 2.2. Приклад матриці підтримки різних версій платформи iOS

Тестування сумісності

Мобільні пристрої різняться залежно від платформи, моделі і версії їх операційної системи. Важливо вибрати таку підмножину пристроїв, яка буде відповідати вашому додатку.

Тестування користувача інтерфейсу

Призначений для користувача досвід є ключовим елементом, при тестуванні програми. Адже наш додаток розробляється саме для кінцевих користувачів. Вам слід якісно перевірити зручність використання додатка, навігацію за його елементами і контент. Тестуйте меню, опції, кнопки, закладки, історію, настройки та навігацію додатка.

Тестування інтерфейсу

Тестування пунктів меню, кнопок, закладок, історії, налаштувань і навігації за додатком.

Тестування зовнішніх чинників

Додатки для мобільних пристроїв не будуть єдиними додатками на пристрої користувача. Разом з вашим додатком будуть встановлені додатки від сторонніх розробників. Можливі десятки таких додатків. Отже, вашому додатку доведеться взаємодіяти з цими сторонніми додатками і переривати роботу різних функцій пристрою, таких, як різні типи мережних підключень, звернення до SD-карті, телефонні дзвінки та інші функції пристрою.

Тестування доступності

Мобільними пристроями можуть користуватися різні люди з обмеженими можливостями. З цієї причини важливо протестувати можливість користуватися послугою людей з дальтонізмом, порушеннями слуху, проблемами людей літнього віку та іншими ймовірними проблемами. Таке тестування є важливою частиною загального тестування юзабіліті.

Етап 6. Тестування продуктивності

Мобільні пристрої надають для додатків меншу місткість пам'яті і меншу доступну потужність процесора, ніж стаціонарні комп'ютери та ноутбуки. З цієї причини у роботі мобільних додатків дуже важлива ефективність використання наданих ресурсів. Вам слід перевірити працездатність тестованого програми, змінивши з'єднання з 2G, 3G, 4G на Wi-Fi, перевірити швидкість відгуку, споживання заряду батареї, стабільність роботи і т. д.

Рекомендується перевіряти додаток на предмет масштабованості застосування і наявності можливих проблем з продуктивністю.

У рамках цього етапу важливо пройти і тестування навантаження мобільного додатка.

Функціональність програми повинна бути повністю протестована. Особливу увагу слід приділити інсталяції оновлень, реєстрації та входу в систему, забезпечення, роботі зі специфічними функціями пристрою і повідомленнями про помилки.

Функціональне тестування мобільного застосування, здебільшого, може бути виконано так само, як Ви виконали б його для будь-якого іншого типу додатка. З цієї причини ми не будемо вдаватися в подробиці цього типу тестування. Однак слід зазначити області, які мають особливе значення для мобільних додатків.

Майте на увазі, що функціональне тестування повинно містити тестування усіх функцій програми, але не повинно бути надмірно зосереджено на якійсь одній функції.

У рамках функціонального тестування, вам слід виконати такі тести:

- тестування процесу інсталяції;
- тестування можливості оновлень додатка;
- експлуатаційне тестування;
- тестування процесу реєстрації та авторизації;
- тестування функцій, специфічних для пристрою;
- тестування відправлення та отримання повідомлень про помилки;
- низькорівневе тестування ресурсів: використання пам'яті, автоматичне вивільнення ресурсів і т. д.
- тестування сервісів: функціонування як в режимі онлайн, так і в автономному режимі.

Етап 7. Атестаційне тестування і тестування безпеки додатків

Безпека і конфіденційність даних мають величезне значення. Користувачі потребують, щоб уся їхня інформація зберігалася безпечно і конфіденційно.

Переконайтеся у тому, що тестований додаток надійно захищений. Виконайте перевірку на можливість упровадження SQL-ін'єкцій, на можливість перехоплення сеансів, аналізу дампів даних, аналізу пакетів і SSL-трафіка.

Дуже важливо перевірити безпеку сховища конфіденційної інформації вашого мобільного додатка і його поведінку відповідно до різних схем дозволів для пристроїв.

Крім перевірки безумовного шифрування імен користувачів і паролів, поставте собі такі запитання:

- Чи є у додатку сертифікати безпеки?
- Чи використовує додаток безпечні мережні протоколи?
- Чи існують будь-які обмеження, наприклад кількість спроб входу в систему до блокування користувачів?

Етап 8. Тестування пристрою

Виконайте тести за тими алгоритмам, які Ви раніше прописали в тестових випадках і сценаріях тестування на всіх визначених для тестування пристроях, в "хмарі" і/або на фізичних пристроях.

Етап 9. Контрольний етап і резюме

Цей етап містить докладне і повне тестування - від ранніх ітеративних етапів тестування до регресійних тестів, які все ще можуть знадобитися для стабілізації роботи програми і виявлення незначних дефектів.

На цьому етапі тестування Ви можете додати для перевірки нові функції і змінити налаштування на ті, яких не буде у фінальній версії.

Після завершення тестування додатка додаткові параметри і функції, додані для перевірки цьому етапі, видаляються, і остаточна версія стає готовою для подання громадськості.

Підсумковий звіт про тестування

Увесь процес тестування мобільних додатків має бути ретельно задокументовано. Перевірте двічі, чи зроблені потрібні записи, і після цього сформуйте свій остаточний звіт про тестування (test summary report).

Цей звіт має містити:

- важливу інформацію, виявлену внаслідок проведених випробувань;
- інформацію про якість проведеного тестування;
- зведену інформацію про якість тестованого мобільного додатка;
- статистику, отриману зі звітів про різні інциденти;
- інформацію про види тестування і час, витрачений на кожен з них.

Слід також звернути увагу у звіті на те, що:

- даний мобільний додаток придатний для використання в тій якості, в якій заявлено;
- відповідає усім критеріям прийнятності функціоналу і якості роботи.

Озброївшись зведенням, керівництво проекту тепер може вирішити, чи готовий мобільний додаток до випуску на ринок.

Тестування мобільних додатків - складна задача. Пристосовуючи ці етапи тестування до кожного з додатків що розробляються, і ретельно виконуючи кожен крок, Ви гарантовано отримаєте повнофункціональний якісний продукт.

3. ПРИНЦИПИ МОБІЛЬНОГО ТЕСТУВАННЯ

У чому особливість мобільного тестування щодо інших видів тестування? Можна виділити такі принципи.

Принцип 1. Постійна мобільність

Головна відмінність мобільного додатка від десктопного полягає у тому, що з мобільним пристроєм Ви пересуваєтеся, марно ловите Wi-Fi в їдальнях ваших вузів, бігаєте, і взагалі по максимуму завантажуєте всі можливі сенсори вашого пристрію. Хтось може сказати, що нерозумно пересуватися по офісу, змінюючи орієнтацію екрана і втрачаючи-відновлюючи Wi-Fi. На це є два контрприклади з життя:

- фатальна помилка (crash) у додатку при спробі відновити його з фонового режиму (background) з попередньою зміною орієнтації екрана;
- фатальна помилка у додатку при "потряхуванні" пристрою в момент виконання цим пристроєм фотозйомки (у додатку для фотографування).

Принцип 2. "Ловля Wi-Fi на живця"

Більша частина сучасних додатків так чи інакше використовує мережу. Далеко не завжди це "повний коннект". Тому обов'язково необхідно тестувати додаток як мінімум чотирма способами:

- позитивний кейс (наявність відмінного постійного зв'язку);
- наявність збоїв у зв'язку;
- відсутність зв'язку;
- утрата зв'язку.

Більшість користувачів навряд чи побачать фатальні помилки вашого додатка, але деякі можуть перестати користуватися додатком, якщо він погано завантажує і зберігає дані користувачів при проблемному зв'язку. На випадок відсутності зв'язку повинні бути відображені хоча б якісь плейсхолдери (аби користувач міг зрозуміти, що програма не може завантажити дані зараз). На останок - утрата зв'язку. Користувачі будуть стикатися з цим постійно, і тут найважливіше - уникнути по можливості втрати даних.

Принцип 3. Переривання

Цей принцип повинен бути головним у списку принципів. Адже переривання наносять істотної шкоди додаткам, при цьому будучи одним з основоположних принципів самих мобільних систем. Під час використання вашого додатку користувач може:

- несподівано отримати дзвінок (смс, нагадування і т. д.);

- закрити програму для того, щоб відкрити якусь іншу на деякий час і повернутися до вашого додатка пізніше;
- відправити девайс у "сон" на деякий час.

Реакцію вашого додатка на ці подразники потрібно перевіряти відразу до функціонального тестування. Тут криється безліч цікавих ситуацій від нешкідливих (трохи "з'їхав" інтерфейс) до фатальних (додаток "падає", дані губляться і т. д.).

Принцип 4. Особливості операційних систем і апаратного забезпечення "заліза"

Даний принцип перебуває тут з явною натяжкою, але все ж уважається, що ОС і "залізо", встановлене в мобільних пристроях, позначається на продуктивності ПЗ набагато сильніше, ніж у десктопних додатках. Чому?

- Android і iOS суттєво відрізняються від десктопних систем. Активності (і їх стек), intent, broadcast-ресивери, маніфест-файл (із зазначенням рівнів доступу), доступ до ресурсів, кеш і дані додатків - усе це та багато іншого потрібно хоча б побіжно вивчити для того, щоб розуміти як ваш додаток працює у цьому конкретному середовищі, що може призвести до його (додатку) відмови або втрати даних. Звідси випливає тестування переходу між станами, тестування для знаходження витоків пам'яті та інше. Усе це необхідно для повноцінного забезпечення якості додатка.
- Прогрес не стоїть на місці, але "залізо" смартфонів все ще сильно обмежено порівняно з настільними системами. Особливо важливо те, скільки ваш додаток займає оперативної пам'яті, за яких умов система автоматично "видалить" його з цієї самої пам'яті, як додаток буде себе вести в цій ситуації.

Принцип 5. Людський фактор

Якщо додаток розрахований на широку аудиторію, будьте готові до того, що купа людей, дуже далеких від розроблення ПЗ, буде вести себе зовсім по-різному з вашим додатком. Тут важливо розуміти, що ми не можемо спрогнозувати всі можливі випадки використання додатка усіма користувачами. Але перевірити, що додаток коректно виконує весь свій функціонал на основних тест кейсах, ми зобов'язані. Непогано також убезпечити себе, "пробігши" по додатку як "користувач-дурник", тикаючи в усі підряд кнопки, відкриваючи й закриваючи активності, не чекаючи завантаження даних і т. д. Проблема "замиленого ока" вирішується шляхом залучення до процесу ваших знайомих, друзів і родичів. Просто дайте їм додаток в руки і подивіться, як вони будуть ним користуватися. Це дасть Вам приблизну інформацію про можливі дефекти додатка.

Тестування мобільних додатків дасть відповіді на такі запитання:

1. Установлення на різних пристроях/ОС

Чи належно встановиться додаток на різних мобільних пристроях, операційних системах (Android, iOS, Windows phone тощо) і різних версіях ОС?

2. Установлення з різних маркетів

Чи належно встановиться додаток, якщо його завантажити з Play Маркет, App Маркет, Android Маркет чи iTunes?

3. Wi-Fi та мережі передачі даних

Чи працюватиме ваш додаток у комп'ютерній мережі і у бездротовій локальній мережі?

4. Умови офлайну (offline)

Що відбуватиметься з додатком у режимі польоту чи в умовах відсутності підключення до інтернету?

5. Вхідний дзвінок

Що станеться, коли вхідний телефонний дзвінок перерве процес установлення чи роботу вашого додатка?

6. Отримання SMS

Що станеться, коли вхідне повідомлення перерве роботу вашого додатка?

7. Перезавантаження телефону

Що станеться, якщо перезавантаження мобільного телефону перерве роботу вашого додатка?

8. Низький заряд батареї

Як ваш додаток зреагує на спливання сповіщення про низький заряд батареї?

9. Недостатньо пам'яті

Як ваш додаток зреагує на сповіщення про те, що в пристрої мало пам'яті?

10. Повертання екрана

Чи реагує ваш додаток на обертання пристрою, відповідно розвертаючи екран вертикально чи горизонтально?

11. Локація

Чи правильно визначається географічне місце розташування, якщо ваш додаток інтегрується із сервісами, що базуються на локації? Чи справно ця опція працює як в мережах даних, так і в бездротових мережах?

12. Зображення

Чи не деформуються ваші малюнки, фотографії, піктограми? Чи задовільно завантажуються всі графічні елементи?

13. Соціальні медіа

Чи встановлена політика конфіденційності та чи налаштовані сповіщення, якщо ваш додаток інтегрується із соцмережами і дозволяє юзерам використовувати їхні акаунти для реєстрації і входу?

14. Швидкодія

Яка мінімальна конфігурація необхідна для запуску програми належним чином? Чи не завадить ваш додаток роботі інших додатків?

15. Безпека

Чи є будь-які вразливі місця, такі, як шкідлива реклама чи будь-які інші кібер-загрози? Чи захищений додаток від них?

4. ОСОБЛИВОСТІ ТЕСТУВАННЯ МОБІЛЬНИХ ДОДАТКІВ

Розглянемо деякі особливості мобільних додатків і вимоги до них.

- До мобільних додатків ставляться підвищені вимоги щодо простоти і зручності використання, якості дизайну, доступності інформації. Це пов'язано з невеликими фізичними розмірами мобільних пристроїв і особливостями взаємодії з використанням жестів і натискань на сенсорні екрани. Вимоги інтерфейсу, призначеного для користувача, взаємодії, вмісту і т. д., зібрані виробниками платформ в керівництва для розробників і досить суворо перевіряються при публікації програм (аж до позбавлення ліцензії розробника за порушення).
- Мобільні додатки часто оновлюються. Причому існують різні шляхи установлення програми (Wi-Fi, 3G/4G, установлення з ПК, на SD, з мобільного маркету, із сайта розробника).
- Локалізація додатків дозволяє відносно простим способом серйозно збільшити цільову аудиторію. Тим паче, що кошти для цього надаються на рівні самої операційної системи пристроїв.
- До мобільних пристроїв висувають підвищені вимоги щодо продуктивності і витрачання ресурсів. Але на відміну від десктопних і веб-додатків, де прагнуть "вичавити" максимум з обладнання, тут намагаються досягти оптимального значення обчислювальної

потужності і витрати ресурсів, для того щоб забезпечити більш тривалий час автономної роботи. У мобільних пристроях також серйозно обмежені ресурси пам'яті, дискового простору і т. п.

- Програма повинна коректно працювати на великому спектрі дозволів екрана і в різних орієнтаціях пристрою.
- Існує велика кількість програмних (версій операційних систем) і апаратних платформ, які надають різні можливості. Замовники хочуть отримати кросплатформні додатки, але навіть у межах однієї платформи важко забезпечити сумісність з усіма варіантами «пристрій/ОС».
- Специфіка використання мобільних пристроїв передбачає велику кількість «помилкових» натискань на клавіші, випадкових впливів на сенсорні екрани (не заблокований телефон в кишені) і програма повинна коректно обробляти такі ситуації.
- Для мобільних пристроїв дуже часті ситуації виникнення переривань у роботі програми. Це можуть бути «природні» причини (вхідний дзвінок або SMS, повідомлення від інших програм, перемикання між програмами, перехід у режим очікування, підключення зарядного пристрою або з'єднання з ПК) і «аварійні» (переривання зв'язку, закінчення заряду акумулятора, падіння пристрою, нестача дискового простору або пам'яті).
- Мобільні телефони можуть бути схильними до ризиків. Через невеликі розміри вони легко можуть бути викрадені фізично, а використання бездротових з'єднань може призвести до втрати даних і несанкціонованого використання мережних ресурсів. Також існують віруси, націлені на використання вразливостей в мобільних протоколах і операційних системах. При цьому на пристроях зазвичай зберігається як персональна (доступ до електронної пошти та соціальних мереж), так і комерційна інформація (мобільний та інтернет-банкінг, доступ до платних он-лайн ресурсів, рахунок мобільного оператора).
- Більшість сучасних мобільних додатків є мережними. Використовують інтернет для різних цілей: оновлення додатка, доступ до інформації та віддалених баз даних, взаємодіють з соціальними мережами та іншими користувачами, збирають і відправляють статику про дії і переваги користувача, завантажують рекламні оголошення.
- У зв'язку з цими та іншими особливостями мобільних платформ і додатків слід особливо звертати увагу на деякі питання при розробленні тестового дизайну. Нижче будуть показані тільки деякі відмінності тестування, тому що більшість видів тестування і тестових методик перегукується з аналогічними видами при тестуванні десктопних і веб-додатків. Особливо це стосується

функціонального тестування.

4.1. Тестування UI / UX / Usability

Під час тестування UI / UX / Usability необхідно звертати на такі аспекти:

- розміри елементів повинні бути такими, щоб користувачеві було комфортно з ними взаємодіяти;
- необхідно тестувати на різних роздільних здатностях екрана (як для мобільних, так і планшетних версій) і в різних орієнтаціях (якщо перемикання орієнтації підтримується програмою);
- кожна дія повинна мати візуальне і можливо інше (аудіо, вібро) підтвердження;
- елементи повинні мати розумний час відгуку при натисканні, скролінгу і т. д.;
- інформація має легко сприйматися (неперевантаженість екранів, мінімум скролінгу);
- наявність підказок, пояснень, відсутність порожніх екранів;
- використання multitouch і багаторазового натискання на один і той же елемент управління;
- інтерфейс не повинен суперечити посібникам;
- необхідно перевіряти, чи використовувалася програма різними групами користувачів (територіальне розміщення, технічні навички, цілі використання і т. д.);
- використання нативних жестів і принципів взаємодії з користувачем.

Android

http://developer.android.com/design/index.html

iOS
https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html

4.2. Тестування локалізації

У додатку використовується багато мов (важко тестувати, багато проблем з перекладом, відображенням, відмінності у культурі різних народів). Переключення мови зазвичай відбувається на рівні операційної системи.

4.3. Тестування на відмову і відновлення

Потрібно тестувати велику кількість різноманітних переривань роботи програми. Необхідно враховувати як дії користувача, так і зовнішні події (переривання зв'язку, відключення живлення, дзвінки).

4.4. Тестування сумісності:

- контроль версій операційної системи;
- перевірка наявності відповідного обладнання (GPS, різні сенсори, апаратна клавіатура, аудіо-відеоконтролери і т. д.);
- сумісність з іншими програмами.

4.5. Тестування продуктивності та ресурсів:

- контроль використання пам'яті і її витоку;
- вільне місце на внутрішніх і зовнішніх носіях;
- кількість інформації, що передається, і швидкість з'єднання;
- навантажувальний і стрес-тестування роботи з віддаленими сервісами.

4.6. Інсталяційне тестування:

- тестування інсталяції;
- тестування видалення;
- збереження і видалення призначених для користувача даних;
- оновлення з різних джерел;
- недостатність ресурсів для установлення або оновлення.

Установлення додатка на SD-карту

Коли додаток установлено з Play Store, Google, він розташовується в пам'яті пристрою. Проте треба перейти в розділ "Налаштування > Додатки" і використовувати перехід до вибору карти пам'яті SD, щоб перемістити додаток.

Перевірки:

- SD-карта відсутня у слоті пристрою під час установлення;
- перезавантажити пристрій, при цьому вийняти SD-карту зі слота та подивитися на поведінку додатка;
- додаток знаходиться у використанні і SD-картку в процесі виймаємо;
- видаляемо частину інформації з SD-картки та дивимось на реакцію додатка;
- установлюємо додаток на зашифровану SD-карту;
- використовуємо ту ж саму карту на іншому пристрої.

Використовуйте Moborobo інструмент для Windows, щоб установлювати додатки безпосередньо на SD-карти.

Оновлення додатка

Установити наявний додаток, зробити необхідні зміни і оновити його. При оновленні програми Android замінює існуючі файли *.jar, якщо є будь-які зміни, і змінює файли бази даних, якщо це необхідно. Усі непотрібні файли з попередньої версії будуть видалені.

Перевірки:

- існуючі налаштування програми можуть бути знищені;
- користувач може отримувати повідомлення про те, що зміни вже були відхилені;
- оновлення може призвести до пошкодження існуючої бази даних;
- необхідні додатку файли можуть бути видалені;

- зашифровані файли користувача можуть бути розшифровані під час оновлення.

4.7. Monkey testing, fuzzy testing, foolproof testing

При перевірці даних видів тестування необхідно виконати такі перевірки:

- моделювання випадкових натискань;
- підтвердження критичних дій (наприклад видалення даних);
- неможливість здійснювати критичні операції випадковими натисканнями або багаторазовим використанням одного елемента управління.

4.8. Тестування безпеки

Очікується, що в майбутньому відсоток мобільних пристроїв, що надають функціональні можливості відкритої платформи, зростатиме. Відкритість цих платформ надає значні переваги всім елементам мобільної системи завдяки можливості гнучкого надання програм і послуг - опцій, які можуть бути встановлені, видалені або оновлені багато разів згідно з потребами і вимогами користувача. Однак з відкритістю з'являється і необмежений доступ до мобільних ресурсів і API-інтерфейсу додатками невідомого або ненадійного походження, що може призвести до заподіяння шкоди користувачеві, налаштуванням, мережі або всього разом узятого, якщо не використовуються відповідні архітектури безпеки і мережні запобіжні заходи. Безпека застосування забезпечується у тій чи іншій формі на більшості мобільних пристроїв з відкритою ОС (Symbian OS, Microsoft, BREW, і т. д.). У 2017 році Google розширив свою програму винагороди за знайдені вразливості. Промислові групи також розробили рекомендації включаючи асоціацію GSM і Open Mobile Terminal Platform (OMTP).

Існує кілька стратегій підвищення безпеки мобільних додатків:

- Складання «білого списку» додатків.
- Забезпечення безпеки транспортного рівня.
- Сувора аутентифікація і авторизація.
- Шифрування даних при записі в пам'ять.
- Пісочниця додатків.
- Надання доступу додатків на рівні API.
- Прив'язка процесів до коду користувача.
- Відсоток взаємодії між мобільним додатком і ОС.
- Вимога авторизації користувача для надання привілейованого/підвищеного доступу з додатком.
- Правильне оброблення сеансу взаємодії.

5. ПРОБЛЕМИ І ТРУДНОЩІ ПРИ ТЕСТУВАННІ МОБІЛЬНИХ ПРИСТРОЇВ

"Перший і найважливіший момент полягає в тому, що тестування мобільних пристроїв займе більше часу, ніж Ви припускаєте, навіть якщо Ви і так допускаєте, що воно займе більше часу." (Peter-Paul Koch).

Мобільні пристрої працюють від акумуляторів, і тому змушені автоматично переходити в режим очікування через декілька хвилин бездіяльності. Це означає, що вам доведеться включати пристрій перед кожним тестуванням, що при одночасному тестуванні декількох пристроїв займає багато часу. Звичайно на багатьох пристроях можна відключити автоматичне блокування (або хоча б зробити час відключення досить великим), але бажано все ж працювати з найпоширенішими серед користувачів налаштуваннями ОС.

Перехід у режим очікування особливо неприємний при використанні деяких автоматизованих систем тестування, що потребують часу для оброблення даних (наприклад, зняття і порівняння скриншотів). Можливо пристрій перейде в режим очікування прямо посеред тесту. Потім з'являється проблема власне набору певного тексту (наприклад, адреси тестованої сторінки). Доведеться акуратно вводити довгі тексти в кілька пристроїв з різними інтерфейсами.

Для тестування в умовах вхідних дзвінків, смс, вам доведеться переставляти SIM-карту з одного пристрою на інший, для цього часто треба виймати акумулятор. Особливо гостро ця проблема стоїть при тестуванні особливостей і послуг мобільних операторів на нестандартних тарифах. Крім того, вставивши SIM-карту, доведеться зачекати, доки телефон не увімкнеться.

Часто доводиться передавати тестовані віджети по Bluetooth, що досить утомливо в умовах розмаїтності інтерфейсів.

Для тестування GPS доведеться озброїтися додатковим інструментарієм від інших розробників і сподіватися, що він працює досить схоже на реальні умови.

Для перевірки слабкого або відсутнього Wi-Fi і 3G/4G-сигналу зазвичай доводиться або споруджувати лабораторію, або використовувати різні хитрощі на кшталт коробочок із фольги.

Створення скриншотів і відео на мобільних пристроях - так само часто нетривіальна робота, особливо якщо тестується телефон, відключений від комп'ютера за умовами тесту або з якихось інших причин. Наприклад, вбудована можливість зняття скриншота екрана на Android-пристроях з'явилася порівняно недавно - з четвертої версії, а про безкоштовний спосіб знімати відеопотік з екрана Apple-пристрої без спеціальної операції jailbreak взагалі мало хто може сказати щось зрозуміле.

Зазвичай всі тести проводяться на одному пристрої, потім - на іншому і т. д. Це не так оптимально, як тестування на всіх пристроях одночасно,

оскільки не дає можливості просто порівняти висновок і виконання тестів на пристроях, проте зважаючи на описані труднощі це - найбільш швидкий спосіб.

Як поліпшити якість праці тестувальника додатків для мобільних пристроїв і позбутися рутини? Очевидно, за допомогою додаткових інструментів - від невеликих додатків і надбудов над SDK до багатофункціональних автоматизованих комбайнів, які здійснюють комплексне тестування.

Захоплення відео з екрана пристрою

Найчастіше можливість записати відео, що відтворює помилки, дуже корисна - це допомагає більш детально описати баг і таким чином заощадити час розробників, а то й уникнути багатьох непорозумінь з багом.

Android screen capture - додаток для передачі відеопотоку з екрана Android-пристроя на монітор комп'ютера. Правда, програма не вміє поки записувати відео - тільки автоматично робити серію скриншотів при зміні екрана. Але ніщо не заважає використовувати десктопні скриншотери з можливістю запису відео. Додаток потребує встановленого Android SDK.

Reflector - платний (від 13 $) інструмент для запису відео з iOS-пристроїв без проводового підключення до десктопу. Працює під Windows і Mac. Він має десятихвилинний тріал. Так само для iPad і iPhone існує додаток Display Recorder, але він постійно то зникає з AppStore, то з'являється. На даний момент пошук в AppStore нічого не дає (зверніть увагу, що Display Recorder HD - це інший додаток, що не має функції запису екрана). Різні джерела називають різні ціни за додатки (від двох до десяти доларів).

5.1. Мобільне кростестування

Основна проблема - потреба у великій кількості пристроїв з різними дозволами екранів, апаратними можливостями і версіями операційних систем. Є три варіанти її вирішення:

- використання реальних пристроїв (якщо не купуються, то беруться в оренду, тестування «Friends & Family»;
- використання емуляторів і симуляторів мобільних пристроїв;
- використання мережних сервісів для отримання доступу до мобільних пристроїв.

5.2. Тестування «Friends & Family»

Використання реальних пристроїв, найбільш переважне для виявлення особливостей функціонування на тій чи іншій платформі, дозволяє

повністю повторити дії користувачів, але є найдорожчим варіантом. Навіть при використанні пристроїв в найближчому оточенні (співробітники, друзі, родина) кількість необхідних конфігурацій може бути дуже велика. При цьому пристрої досить швидко застарівають і потрібно постійно купувати нові.

6. ТЕСТУВАННЯ ЗА ДОПОМОГОЮ ЕМУЛЯТОРІВ І СИМУЛЯТОРІВ

Емулятори та симулятори є найбільш поширеними інструментами для тестування мобільних додатків.

Переваги:

- дозволяють тестувати додаток для різних розмірів екранів, версій ОС і т. п.;
- доступність (зазвичай безкоштовні в складі засобів розроблення);
- можливість перегляду коду, покрокового виконання, перегляду логів і значень змінних у середовищі розроблення;
- вища швидкість тестування за рахунок більших ресурсів ПК і зручнішого введення з клавіатури;
- просте отримання скриншотів.

Недоліки:

- не відповідають призначеним для користувача пристроям за ресурсами і характеристиками;
- не дозволяють моделювати всі дії (переривання живлення, зв'язку, мультитач);
- існують не для всіх пристроїв;
- деякі режими роботи (наприклад, дзвінки) недоступні;
- немає роботи з реальною «начинкою» мобільних пристроїв, а тільки з їх моделями;
- немає гарантії, що помилка, яка з'являється на емуляторі, буде і на пристрої, і навпаки.

Нижче показано вигляд емулятора Android і наведено посилання на інформацію щодо емуляторів і симуляторів різних платформ (рис. 6.1).

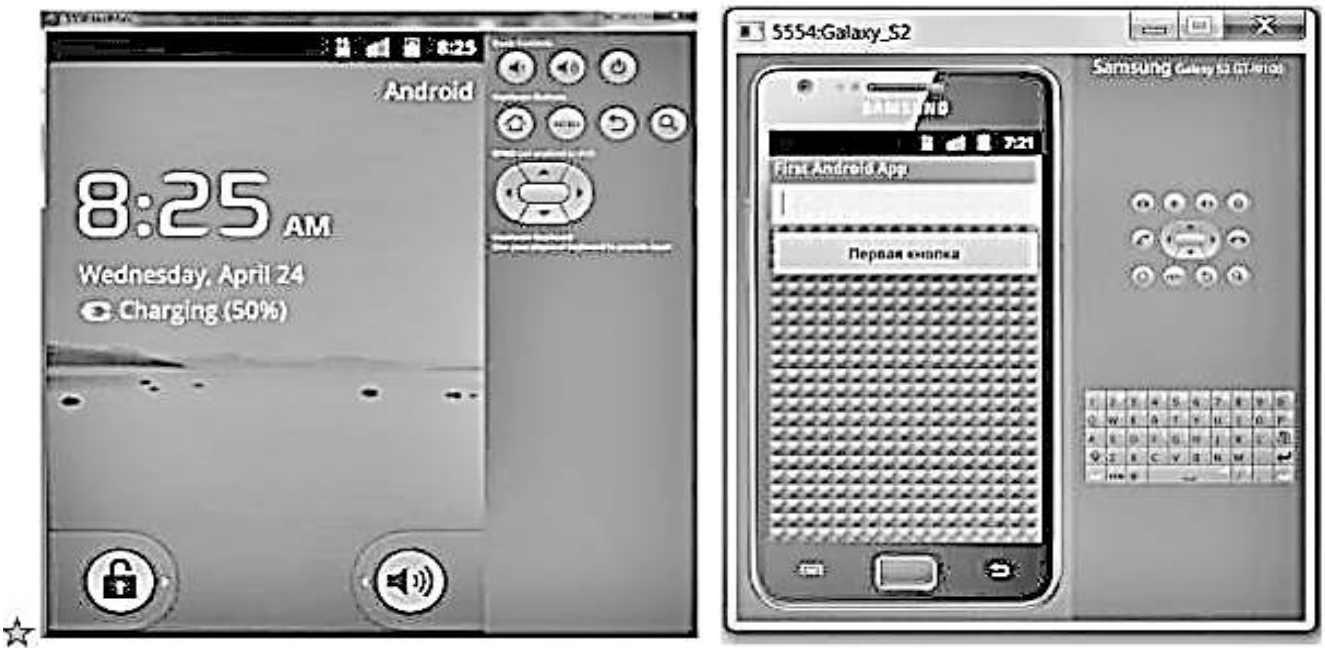


Рис. 6.1. Приклад емуляторів

http://developer.android.com/tools/devices/emulator.html

https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/iOS_Simulator_Guide/Introduction/Introduction.html

6.1. Запуск програми на емуляторі Android

Android Емулятор імітує Android пристрії на вашому комп'ютері, так що Ви можете перевірити свої додатки на різних пристроях і різних рівнях API.

Емулятор забезпечує майже всі можливості реального пристрою Android. Ви можете імітувати вхідні телефонні дзвінки і текстові повідомлення, указувати місце знаходження пристрою, імітувати різні швидкості мережі, датчиків обертання та інших апаратних засобів, доступ до Google Play Store і багато іншого.

Тестування програми на емуляторі дещо швидше і простіше, ніж робити це на фізичному пристрої. Наприклад, Ви можете передавати дані швидше, ніж емулятор на пристрій, підключений через USB.

Емулятор поставляється з попередньо встановленими конфігураціями для різного Android телефону, планшета, перевстановлення ОС і пристроїв Android TV.

Ви можете використовувати емулятор вручну за допомогою графічного інтерфейсу користувача і програмно за допомогою командного рядка і консолі емулятора.

Android Emulator має додаткові вимоги, крім основної системної вимоги для Android Studio:

- SDK Tools 26.1.1 або вище;
- 64-бітний процесор;
- підтримка CPU з UG (необмежений гість);
- HAXM 6.2.1 або пізнішої версії (HAXM 7.2.0 або пізнішої версії).

Використання апаратного прискорення має додаткові вимоги до ОС Windows і Linux:

- процесор Intel на Windows або Linux: процесор Intel з підтримкою Intel VT-x, Intel EM64T (Intel 64) і Execute Disable (XD) функціональність Bit;
- процесор AMD на Linux: AMD процесор з підтримкою віртуалізації AMD (AMD-V) і SIMD Extensions 3 (SSSE3);
- процесор AMD на Windows: Android Studio 3.2 або вище і Windows.

Для роботи з Android 8.1 (рівень API 27) і вищими версіями системи повинна бути веб-камера з можливістю захоплення 720p кадрів.

Версії для 32-бітових систем Windows

Про Android Emulator для 32-розрядних систем Windows негативно висловлювалися в червні 2019. Однак підтримка 32-розрядної ОС Windows емулятора триває до червня 2020 року, в тому числі для виправлення критичних помилок, але при цьому не буде додано ніяких нових функцій.

Якщо Ви використовуєте емулятор на 32-бітній системі Windows, Ви маєте планувати міграцію на 64-бітну систему Windows.

Якщо Ви використовуєте емулятор на 32-бітній системі Windows, Ви можете використовувати SDK менеджер, встановивши останню версію емулятора для 32-бітної Windows.

Встановлення емулятору

Щоб установити Android Emulator, виберіть компонент Android Emulator на вкладці SDK Tools диспетчера SDK.

Кожен екземпляр Android Emulator використовує Android Virtual Drive (AVD), який емулює версії Android і апаратні характеристики модельованого пристрою. Для того щоб ефективно протестувати додаток, Ви маєте створити AVD, для моделі кожного пристрою, на якому ваш додаток призначений для запуску. Для створення і управління AVD використовуйте AVD-менеджер,

Кожна функція AVD є незалежним пристрієм з власним зберіганням призначених для користувача даних, SD-карти і т. д. За замовчуванням, емулятор зберігає призначені для користувача дані, дані SD-карти і кеш у директорії, специфічній для цього AVD. Коли Ви запускаєте емулятор, він завантажує призначені для користувача дані і дані SD-карти з каталогу AVD.

Запуск програми на емуляторі Android

Ви можете запустити додаток з проекту Android Studio або можете запустити додаток, уже встановлений на емуляторі Android, як би Ви запустити будь-який додаток на пристрої (рис. 6.2).

Щоб запустити Android Emulator і запустити додаток у вашому проекті необхідно:

- в Android Studio створити AVD, щоб емулятор можна було використовувати для установки і запуску програми,
- на панелі інструментів вибрати AVD, в меню якого вибрати цільовий пристрій, на якому Ви хочете запустити додаток, що тестується.
- натисніть кнопку Виконати  .

Якщо з'являється повідомлення про помилку або попередження у верхній частині діалогового вікна, натисніть на посилання, щоб виправити цю проблему або отримати додаткову інформацію.

Деякі помилки необхідно виправити перш ніж продовжувати роботу, наприклад певні апаратні помилки Accelerated Execution Manager (Intel HAXM).

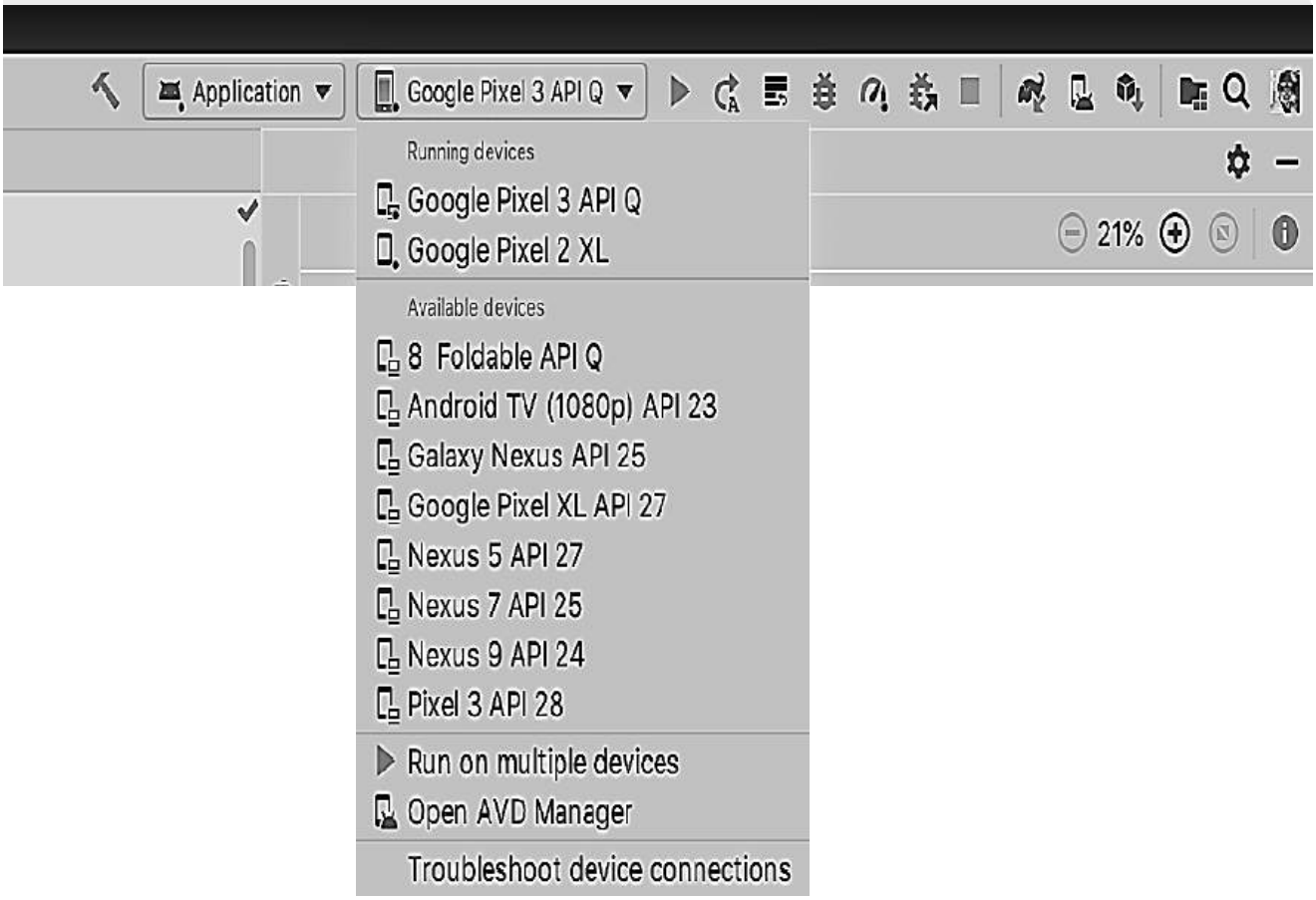


Рис. 6.2. Головне меню Android Emulator

Для MacOS, якщо Ви бачите попередження: "Немає DNS-сервера", "Знайдені помилки при запуску емулятора", то перевірте, чи є файл /etc/resolv.conf. Якщо у вас немає цього файла, введіть наступну команду у вікні терміналу:

ln -s /private/var/run/resolv.conf /etc/resolv.conf

Запуск Android Emulator без першого запуску додатка

Щоб запустити емулятор, слід:

- відкрити диспетчер AVD;
- двічі натиснути "A" або вибрати команду Виконати  , з'явиться Android Emulator.

Хоча емулятор працює, Ви можете запустити проекти Android Studio і вибрати емулятор як цільовий пристрій. Ви також можете перетягнути один або кілька файлів *.apk на емулятор, щоб установити їх, а потім запустити.

Установлення і додавання файлів

Щоб встановити файл *.apk на емуляторі, перетягніть цей файл на екран емулятора. З'явиться діалогове APK Installer. Після завершення установки Ви можете переглядати програми у списку програм.

Файл поміщається в розділ: /SDCARD/downloads/catalog. Ви можете переглянути файл з Android Studio за допомогою File Explorer або знайти його з пристрою за допомогою завантаження або файлів програми - залежно від версії пристрою.

Скриншот зберігає весь стан пристрою в той час, коли він був збережений - в тому числі налаштування ОС, стан додатка і призначені для користувача дані. Ви можете повернутися до збереженого стану системи шляхом завантаження скриншоту щоразу, коли Вам це необхідно. Запуск віртуального пристрою при завантаженні скриншоту дуже схожий на пробудження фізичного пристрою зі стану сну, на відміну від завантаження його зі стану вимкнення живлення.

Для кожного AVD Ви можете мати один швидкий скриншот завантаження і будь-яку кількість загальних скриншотів.

Найпростіший спосіб скористатися скриншотами полягає у використанні швидкого їх завантаження.

Перший раз, коли починаються завантаження скриншоту, емулятор повинен виконати "холодне" перезавантаження, так само, як при ввімкненні пристрою. Якщо опція Quick Boot увімкнена, то всі наступні завантаження зазначеного скриншоту відбудуться швидко.

Скриншоти дійсні для зображення системи, конфігурації AVD і функцій емулятора, з якими вони зберігаються. Коли Ви вносите зміни у будь-яку з цих областей, усі скриншоти такої AVD стають недійсними. Будь-яке оновлення Android Emulator, скриншоту системи або настройки AVD скидає збережений стан AVD, тому наступного разу, коли Ви запустите AVD, він повинен виконати "холодне" перезавантаження.

Більшість елементів управління для збереження, завантаження і управління скриншотами в панелі Snapshots реалізовано в емуляторі за допомогою вікна Розширені налаштування (Extended Control) (рис. 6.3).

Ви також можете керувати скриншотами в емуляторі за допомогою командної строки.

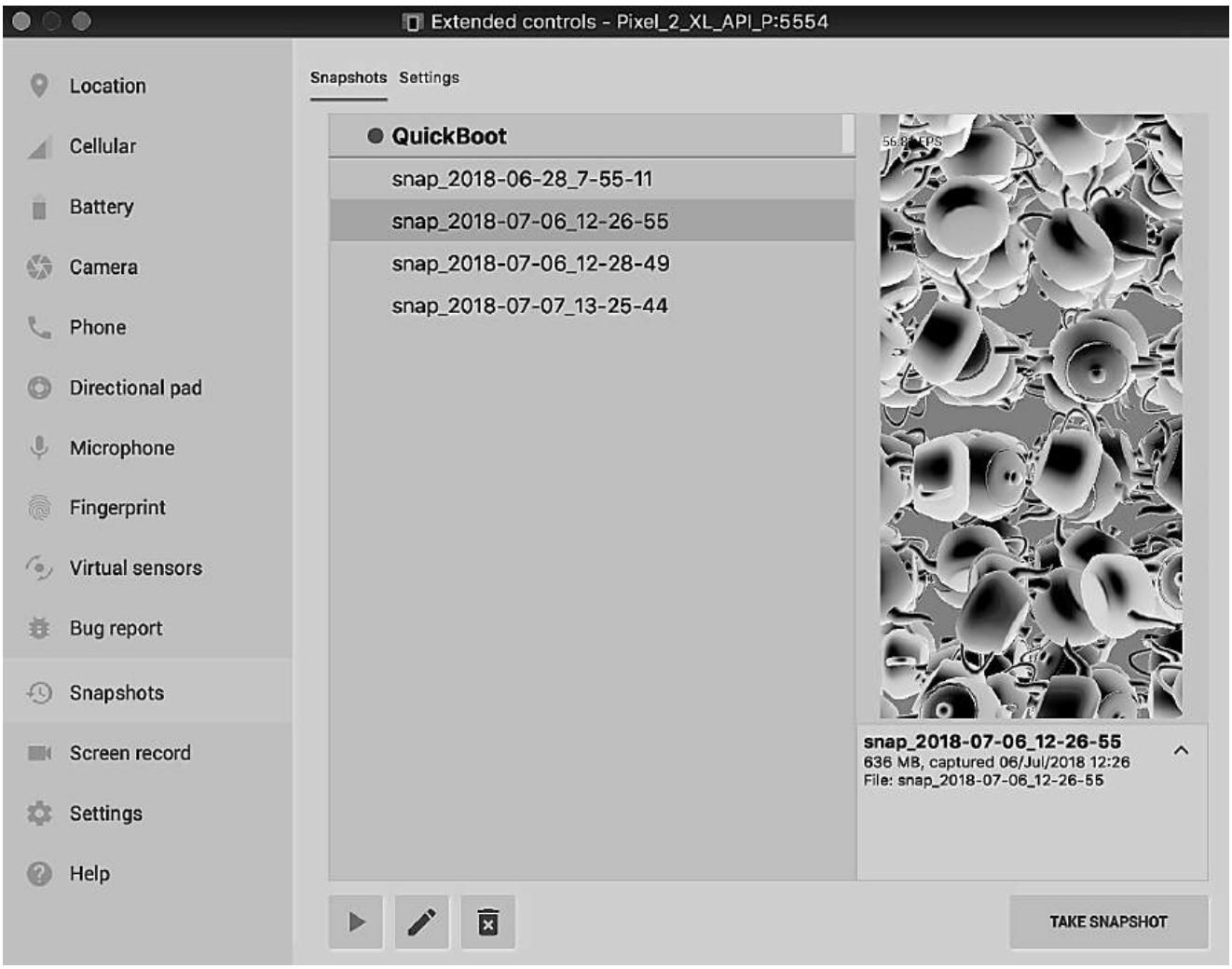


Рис. 6.3. Управління скриншотами в панелі Snapshots

Налаштування швидкого збереження завантаження скриншотів

Для того щоб контролювати, чи буде автоматично зберігатися скриншот в емуляторі для відкритої AVD при виході, використовуйте стан Save з меню емулятора на вкладці Параметри в категорії Скриншоти у вікні Розширені налаштування. Налаштування має такі параметри:

- **Так -** завжди зберегти скриншот AVD при закритті емулятора. Це значення за замовчуванням.
- **Немає -** не зберегати скриншот AVD при закритті емулятора.
- **Запитати**: Запитувати, чи зберегти знімок AVD при закритті емулятора.

Ваш вибір стосується тільки поточного відкритого AVD.

Якщо Ви не вибираєте в меню параметр Yes у стані Save для швидкого завантаження скриншотів, Ви можете використовувати кнопку

Save Now у меню, щоб зберегти швидке завантаження скриншотів у будь-який час.

Ви не можете зберігати скриншоти, коли AVD не запущено.

Зберегання загальних скриншотів

У той час як Ви можете мати тільки один швидкий скриншот, завантаженний для кожного AVD, Ви можете мати декілька загальних знімків для кожного AVD.

Для того щоб зберегти загальний знімок, відкрийте емулятор, перейдіть у вікно Розширені засоби управління, виберіть категорію Скриншоти і натисніть кнопку Сфотографувати в нижньому правому куті вікна.

Щоб змінити назву і опис обраного скриншоту, натисніть правку  . Ця кнопка знаходиться у нижній частині вікна.

Видалення скриншоту

Щоб вручну видалити скриншот, відкрийте емулятор і перейдіть у вікно Розширені засоби управління, виберіть категорію Знімки, виберіть зображення і натисніть кнопку видалення  . Кнопка знаходиться в нижній частині вікна.

Ви також можете вказати, чи хочете Ви, щоб емулятор автоматично видаляв знімки, коли вони стають недійсними, наприклад, коли зміняться настройка або версія емулятора AVD. За замовчуванням емулятор запитає вас, чи Ви хочете видалити неприпустимі знімки. Ви можете змінити цю настройку за допомогою меню Delete на вкладці Налаштування панелі Snapshots.

Завантаження скриншоту

Для того щоб завантажити скриншот в будь-який момент, треба відкрити емулятор, перейти у вікно Розширені засоби управління, вибрати категорію Скриншоти, вибрати зображення і натиснути Завантаження 
Ця кнопка знаходиться в нижній частині вікна.

В Android Studio 3.2 і вище кожна конфігурація пристрою має управління опцією завантаження в розширених налаштуваннях у діалозі

Конфігурація віртуального пристрою, за допомогою якого можна вказати, який знімок треба завантажувати при запуску AVD.

Відключення швидкого завантаження

Якщо Ви хочете відключити Quick Boot, щоб ваш AVD завжди виконував "холодне" перезавантаження, виконайте такі дії:

- виберіть Інструменти > AVD Manager і натисніть кнопку Змінити цей AVD ;
- натисніть Показати додаткові налаштування і перейдіть до емулювання Performance;
- виберіть "холодне" завантаження;
- "холодне" завантаження неактивне.

Замість того щоб повністю відключити Quick Boot, Ви можете відключити "холодне" завантаження, тільки один раз натиснувши пункт Cold Boot Now з випадаючого меню AVD (рис. 6.4, табл. 6.1).

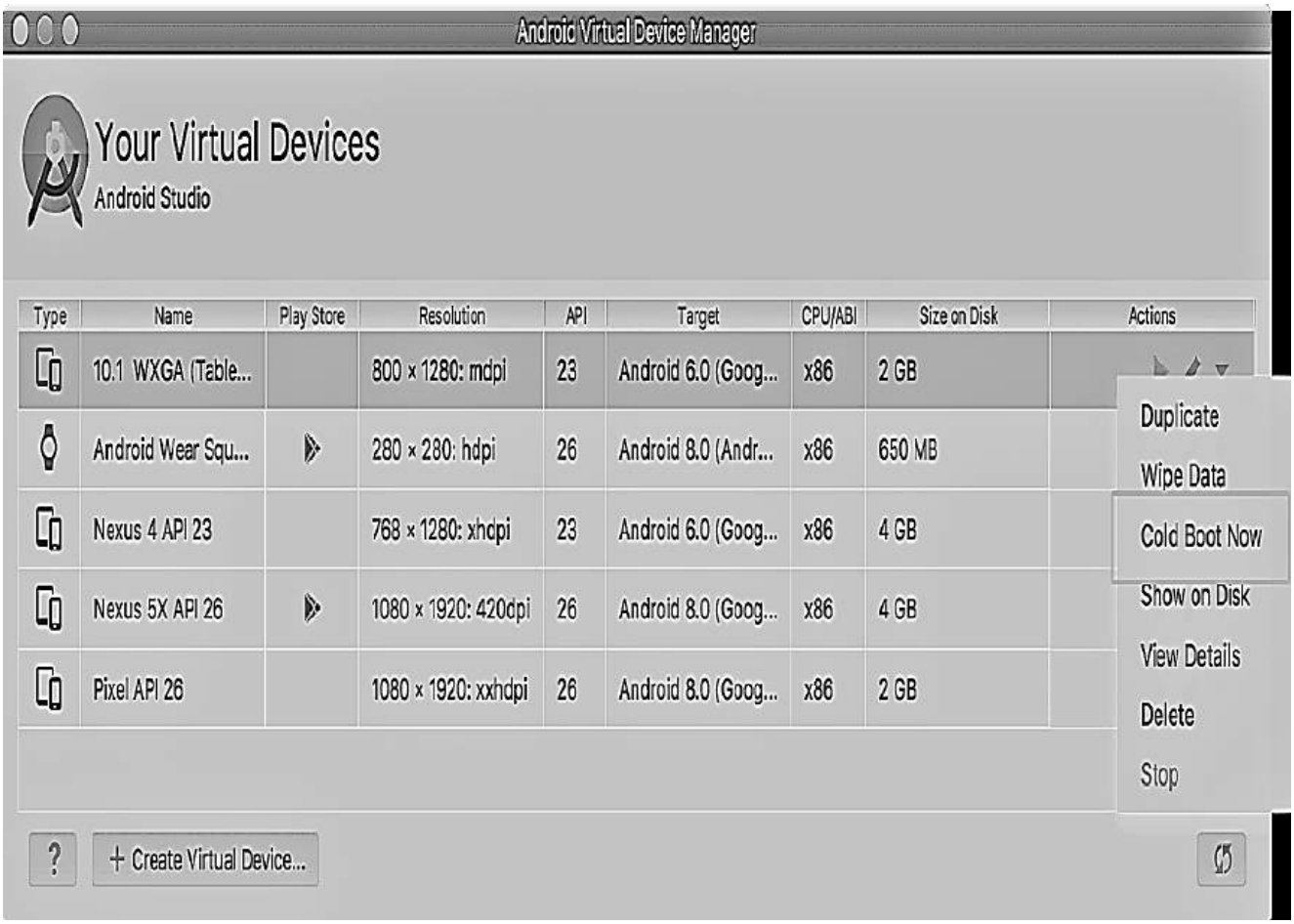


Рис. 6.4. Інтерфейс AVD Manager

Таблиця 6.1

Жести для навігації екрана емулятора

Особливість	Опис
Свайп екрана (Swipe the screen)	Наведіть покажчик миші на екран, натисніть і утримуйте першу кнопку миші, проведіть пальцем по екрану, а потім відпустіть
Перетягування елемента (Drag an item)	Виберіть точку елемента на екрані, натисніть і утримуйте першу кнопку миші, перемістіть елемент, а потім відпустіть
Торкання екрана (Tap, touch)	Наведіть покажчик миші на екрані, натисніть першу кнопку миші, а потім відпустіть. Наприклад, Ви можете натиснути на текстове поле, щоб почати друкувати в ньому, виберіть додаток або натисніть кнопку
Подвійне торкання (Double tap)	Наведіть покажчик миші на екрані, натисніть першу кнопку миші в два рази швидше, а потім відпустіть
Натискання і утримання елемента	Виберіть точку елемента на екрані, натисніть першу кнопку миші та утримуйте її, а потім відпустіть. Наприклад, Ви можете відкрити опції для пункту
Уведення даних (Type)	Уведіть дані в емулятор за допомогою клавіатури комп'ютера або за допомогою клавіатури, яка з'являється на екрані емулятора. Наприклад, можна ввести в текстове поле дані після того, як Ви вибрали його
Пінч-жест (Pinch)	Одночасно натисніть двома пальцями розташовані поблизу або на деякій відстані точки на екрані з подальшим їх зближенням. Жест завершується підняттям пальців над екраном. Очевидно, в результаті такої механічної дії значно зменшується масштаб. У iOS цей жест може використовуватися для закриття фотографії та виходу з альбому
Вертикальний свайп (Vertical swipe)	Відкрийте вертикальне меню на екрані і використайте колесо прокрутки (колесо миші) для переміщення по пунктах меню, доки не з'явиться той пункт, який Вам потрібен. Натисніть на пункт меню, щоб вибрати його

Проблеми та методи їх усунення:

- Скриншоти не працюють з Android 4.0.4 (рівень API 15) або нижче.
- Скриншоти не працюють з системою ARM зображень для Android 8.0 (рівень API 26).
- Якщо емулятор не завантажує скриншоти, виберіть процедури Now для AVD у диспетчері AVD і повідомте про помилку.

- Скриншоти не знадійні, якщо візуалізація програмного забезпечення ввімкнена. Якщо скриншоти не працюють, натисніть кнопку Змінити цю AVD у менеджері та змінить графіки AVD апаратного забезпечення.
- Завантаження або збереження скриншоту є інтенсивною роботою для оперативної пам'яті. Якщо у вас немає достатньої кількості вільної оперативної пам'яті при завантаженні або збереженні скриншотів, операційна система може перенести вміст оперативної пам'яті на жорсткий диск, що може значно сповільнити роботу. Тому краще заздалегідь вивільнити оперативну пам'ять. Використовуйте покажчик комп'ютерної миші, щоб імітувати ваш палець на сенсорному екрані; виберіть пункти меню і поле введення; натисніть кнопки і елементи управління. За допомогою комп'ютерної клавіатури для введення символів уведіть ярлики емулятора.

Виконання спільніх дій в емуляторі

Для того щоб виконати загальні дії з емулятором, використайте панель з правого боку, як описано в табл. 6.2.

Ви можете використовувати поєднання клавіш для виконання багатьох спільних дій в емуляторі. Для отримання повного списку ярликів в емуляторі натисніть F1 (Ctrl + / на Mac), щоб відкрити панель довідки у вікні Розширені засоби управління.

Таблиця 6.2

Спільні дії в емуляторі

Кнопка управління	Опис
Закрити	Закрити емулятор
Мінімізувати	Згортання вікна емулятора
Зміна розміру	Зміна розміру емулятора виконується, як і для будь-якого іншого вікна операційної системи. Емулятор підтримує співвідношення сторін відповідно до налаштувань вашого пристрою
Живлення	Натисніть, щоб включити або виключити екран. Натисніть і утримуйте, щоб увімкнути або вимкнути пристрій

Продовження табл. 6.2

Кнопка управління	Опис
Збільшення гучності 	Натисніть і перемістить повзунок угору для збільшення гучності
Зменшення гучності 	Натисніть та перемістить повзунок униз для зменшення гучності
Поворот вліво 	Поворот пристрою на 90 градусів проти годинникової стрілки
Поворот вправо 	Поворот пристрою на 90 градусів за годинниковою стрілкою
Зробити скриншот 	Натисніть, щоб зробити знімок екрана пристрою
Вхід у режим масштабування 	Натисніть на значок масштабування. Для того щоб вийти з режиму збільшення, натисніть кнопку ще раз. Збільшення і зменшення масштабу в режимі масштабування: • натисніть лівою кнопкою миші на екрані, щоб збільшити масштаб з кроком у 25 %, до максимуму приблизно в два рази більше роздільної здатності екрана віртуального пристрою • натисніть правою кнопкою миші, щоб зменшити масштаб • натисніть лівою кнопкою миші і перетягніть мишу, щоб вибрати область, яку необхідно збільшити • натисніть правою кнопкою миші і перетягніть поле вибору для скидання збільшення за замовчуванням Для переміщення в режимі масштабування утримуйте кнопку Control (Command на Mac) і одночасно натискайте клавіші зі стрілками на клавіатурі

Закінчення табл. 6.2

Кнопка управління	Опис
Назад 	Повернення до попереднього екрана або закриття діалогового вікна, меню параметрів, панелі сповіщень або екранної клавіатури
Головна сторінка 	Повернення до головного вікна
Огляд  (Останні програми)	Натисніть, щоб відкрити список мініатюрних зображень додатків, з якими Ви нещодавно працювали. Щоб відкрити програму, натисніть на неї. Щоб видалити зменшене зображення зі списку, проведіть пальцем вліво або вправо. Ця кнопка не підтримує "зносу" ОС
Складка 	Для складних пристроїв складіть пристрій, щоб відобразити інформацію на зовнішньому екрані
Розкривання 	Для складних пристроїв розкрийте пристрій для відображення інформації на внутрішньому (великому) екрані
Меню	Натисніть Ctrl+M (Ctrl+M на Mac), щоб імітувати кнопку меню
Більше 	Натисніть, щоб отримати доступ до інших функцій і налаштувань

Запис екрана

Ви можете записувати відео і аудіо з Android Emulator і зберігати запис у форматі WebM або анімованому файлі GIF.

Управління записом екрана знаходиться у вікні Розширені елементи управління.

Для початку запису екрана натисніть на кнопку Старт на вкладці Запис екрана. Щоб зупинити запис, натисніть кнопку Зупинити запис.

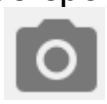
Елементи управління для відтворення і збереження записаного відео знаходяться в нижній частині вкладки Запис екрана. Щоб зберегти відео, виберіть WebM або GIF з меню в нижній частині вкладки і натисніть кнопку Зберегти.

Крім того, можна записувати і зберігати записи екрана з емулятора, використовуючи таку команду в командному рядку:

adb emu screenrecord start --time-limit 10 [path to save video]

/sample_video.webm

Скриншоти

Для того щоб зробити скриншот віртуального пристрою, натисніть на кнопку Скриншот  .

Емулятор створює PNG-файл з ім'ям *Screenshot_yyyymmdd-hhmmss.png* використовуючи рік, місяць, день, годину, хвилину і секунду захоплення. Наприклад, Screenshot_20190219-145848.png.

За замовчуванням скриншот зберігається на робочому столі комп'ютера. Щоб змінити місце розташування, в якому зберігаються скриншоти, використовуйте категорію Screenshot у вікні Налаштування в емуляторі.

Ви можете також робити скриншоти з командного рядка за допомогою будь-якої з таких команд:

- *screenrecord screenshot [destination-directory]*
- *adb emu screenrecord screenshot [destination-directory]*

Камера віртуальної сцени і ARCORE

Ви можете використовувати камеру віртуальної сцени у віртуальному середовищі, щоб експериментувати з доповненою реальністю (AR) для додатків, зроблених за допомогою ARCORE.

При використанні емулятора з додатком камери Ви можете імпортувати зображення у форматі PNG або JPEG для використання у віртуальній сцені. Для того щоб вибрати зображення для використання у віртуальній сцені, натисніть кнопку Додати зображення в Camera і перейдіть у вкладку Віртуальні зображення сцени в розширеному вікні елементів управління. Ця функція може бути використана для імпорту призначених для користувача зображень, таких, як QR-код.

Перевірка загальних дій AR з макросами

Ви можете значно скоротити час, необхідний для перевірки загальних AR дій, використовуючи встановлені макроси в емуляторі. Наприклад, Ви можете використовувати макрос, щоб скинути всі датчики пристрою за замовчуванням.

Перед використанням макросів виконайте дії, описані в розділі Запуск AR, в емуляторі Android налаштуйте камеру віртуальної сцени для вашого додатка, запустіть додаток на емуляторі і оновіть ARCORE. Потім виконайте такі дії, щоб використовувати емулятор макросів:

- щоб з емулятором працювали і додатки, підключені до ARCORE, натисніть Далі на панелі емулятора;
- виберіть пункт Record and Playback > Macro Playback;
- виберіть макрос, який Ви хочете використовувати, а потім натисніть кнопку відтворення.

Під час відтворення Ви можете перервати макрос, натиснувши кнопку Стоп.

Розширені засоби управління, налаштування і допомога

Використовуйте розширені засоби управління для передачі даних, зміни налаштувань пристрою, додатків управління і багато іншого. Щоб відкрити вікно Розширені засоби управління, натисніть кнопку Далі  на панелі емулятора.

Ви можете використовувати поєднання клавіш для виконання багатьох з цих завдань. Для отримання повного списку ярликів в емуляторі натисніть F1 (Ctrl + / на Mac), щоб відкрити панель довідки. В англомовній спільноті є додаткові пояснення щодо розширених засобів управління (з метою мінімізації викривлень перекладу подаємо на мові оригіналу в авторському вигляді).

Extended controls details

Feature	Description
Location	The emulator lets you simulate "my location" information: the location where the emulated device is currently located. For example, if you click My Location  in Google Maps and then send a location, the map shows it. To send a GPS location: 1. Select Decimal or Sexagesimal. 2. Specify the location. In decimal mode, enter a Latitude value in the range -90 ... +90 degrees and a Longitude value in the range -180 ... +180 degrees. In sexigesimal mode, enter a three-part Latitude value in the range -90 ... +90 degrees, 0 ... 59 minutes, and 0 ... 60 seconds. Enter a Longitude value in the range -180 ... +180 degrees, 0 ... 59 minutes, and 0 ... ... 60 seconds. For the latitude, - indicates south and + indicates

north; for the longitude, - indicates west and + indicates east. The + is optional.

Optionally specify an Altitude value in the range -1000 ... +10000 meters.

3. Click Send.

To use geographic data from a GPS exchange format (GPX) or Keyhole Markup Language (KML) file:

1. Click Load GPX/KML.
2. In the file dialog, select a file on your computer and click Open.
3. Optionally select a Speed. The speed defaults to the Delay value (Speed 1X). You can increase the speed by double (Speed 2X), triple (Speed 3X), and so on.
4. Click Run  .

Cellular

The emulator lets you simulate various network conditions. You can approximate the network speed for different network protocols, or you can specify Full, which transfers data as quickly as your computer allows. Specifying a network protocol is always slower than Full. You can also specify the voice and data network status, such as roaming. The defaults are set in the AVD.

Select a Network type:

- GSM: Global System for Mobile Communications
- HSCSD: High-Speed Circuit-Switched Data
- GPRS: Generic Packet Radio Service
- EDGE: Enhanced Data rates for GSM Evolution
- UMTS: Universal Mobile Telecommunications System
- HSPDA: High-Speed Downlink Packet Access
- LTE: Long-Term Evolution
- Full (default): Use the network as provided by your computer

Select a Signal strength:

- None
- Poor
- Moderate (default)
- Good
- Great

Select a Voice status, Data status, or both:

- Home (default)

- Roaming
- Searching
- Denied (emergency calls only)
- Unregistered (off)

Battery

You can simulate the battery properties of a device to see how your app performs under different conditions. To select a Charge level, use the slider control.

Select a Charger connection value:

- None
- AC charger

Select a Battery health value:

- Good (default)
- Failed
- Dead
- Overvoltage
- Overheated
- Unknown

Select a Battery status value:

- Unknown
- Charging (default)
- Discharging
- Not charging
- Full

Phone

The emulator lets you simulate incoming phone calls and text messages.

To initiate a call to the emulator:

1. Select or type a phone number in the From field.
2. Click Call Device.
3. Optionally click Hold Call to put the call on hold.
4. To end the call, click End Call.

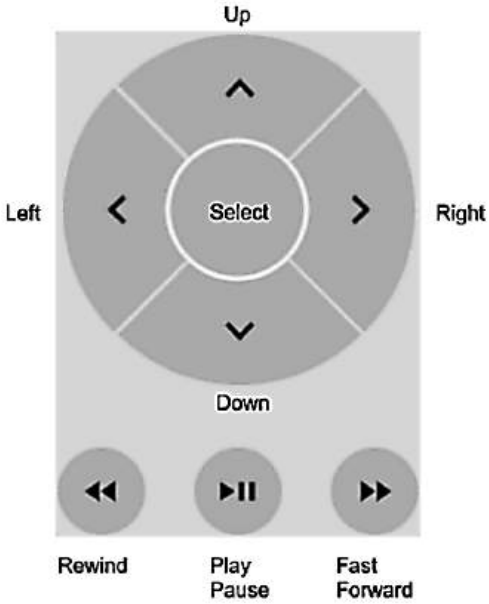
To send a text message to the emulator:

1. Select or type a phone number in the From field.
2. Type a message in the SMS message field.
3. Click Send Message.

Directional Pad

If the AVD has the directional pad enabled in the hardware profile, you can use the directional pad controls with the emulator. However, not all devices can support the directional pad; for example, an Android watch. The buttons simulate the

following actions:



Fingerprint

This control can simulate 10 different fingerprint scans. You can use it to test fingerprint integration in your app. This feature is disabled for Android 5.1 (API level 22) and lower, and for Wear OS.

To simulate a fingerprint scan on the virtual device:

1. Prepare an app to receive a fingerprint.
2. Select a Fingerprint value.
3. Click Touch Sensor.

Virtual sensors > Accelerometer

This control lets you test your app against changes in device position, orientation, or both. For example, you can simulate gestures such as tilt and rotation. The accelerometer doesn't track the absolute position of the device: it just detects when a change is occurring. The control simulates the way accelerometer and magnetometer sensors would respond when you move or rotate a real device.

You must enable the accelerometer sensor in your AVD to use this control.

The control reports TYPE_ACCELEROMETER events on the x, y, and z axis. These values include gravity. For example, if the device is suspended in outer space, it would experience zero acceleration (all of x, y, and z will be 0). When the device is on Earth and laying screen-up on top of a table, the acceleration is 0, 0, and 9.8 because of gravity.

The control also reports TYPE_MAGNETIC_FIELD events, which measure the ambient magnetic field on the x, y and z axis in microteslas (μT).

To rotate the device around the x, y, and z axes,

select Rotate and do one of the following:

- Adjust the Yaw, Pitch, and Roll sliders and observe the position in the upper pane.
- Move the device representation in the upper pane and observe the Yaw, Pitch, and Roll and how the resulting accelerometer values change. To move the device horizontally (x) or vertically (y), select Move and do one of the following:
- Adjust the X and Y sliders and observe the position in the upper pane.
- Move the device representation in the upper pane and observe the X and Y slider values and how the resulting accelerometer values change. To position the device at 0, 90, 180, or 270 degrees:
- In the Device rotation area, select a button to change the rotation.

As you adjust the device, the Resulting values fields change accordingly. These are the values that an app can access.

You can import the AccelerometerPlay app to try out the Accelerometer control. Select File > New > Import Sample and select the app in the dialog. This app is showcased in the emulator video on this page.

Virtual sensors > Additional sensors

The emulator can simulate various position and environment sensors. It lets you adjust the following sensors so you can test them with your app:

- Ambient temperature: This environmental sensor measures ambient air temperature.
- Magnetic field: This position sensor measures the ambient magnetic field on the X, Y, and Z axes, respectively. The values are in microteslas (μT).
- Proximity: This position sensor measures the distance from an object; for example, it can notify a phone that a face is close to it to make a call. The proximity sensor must be enabled in your AVD to use this control.
- Light: This environmental sensor measures illuminance. The values are in lux units.
- Pressure: This environmental sensor measures ambient air pressure. The values are in millibar (hPa) units.

Relative Humidity: This environmental sensor measures

ambient relative humidity.

Settings > General	>	• Emulator window theme: Select Light or Dark. • Send keyboard shortcuts to: By default, some keyboard combinations will trigger emulator control shortcuts. If you're developing an app that includes keyboard shortcuts, such as one targeted at devices with Bluetooth keyboards, you can change this setting to send all keyboard input to the virtual device, including input that would be a shortcut in the emulator. • Screenshot save location: Click the folder icon to specify a location to save screenshots of the emulator screen. • Use detected ADB location: If you're running the emulator from Android Studio, you should select this setting (the default). If you run the emulator from outside Android Studio and want it to use a specific adb executable, deselect this option and specify the SDK Tools location. If this setting is incorrect, features such as screenshot capture and drag-and-drop app installation won't work. • When to send crash reports: Select Always, Never, or Ask. • Show window frame around device: By default, emulators with device skin files are shown without a surrounding window frame.
Settings > Proxy	>	By default, the emulator uses the Android Studio HTTP proxy settings, but this screen allows you to manually define an HTTP proxy configuration for the emulator. For more information, see Using the emulator with a proxy.
Settings > Advanced	>	• OpenGL ES renderer: Select the graphics acceleration type. (This is equivalant to the -gpu command line option). • Autodetect based on host: Let the emulator choose hardware or software graphics acceleration based on your computer setup. It checks if your GPU driver matches a list of known faulty GPU drivers, and if it does, the emulator disables graphics hardware emulation and instead uses the CPU. • ANGLE: (Windows only.) Use ANGLE Direct3D to render graphics in software.

- SwiftShader: Use SwiftShader to render graphics in software.
- Desktop native OpenGL: Use the GPU on your host computer. This option is typically the fastest. However, some drivers have issues with rendering OpenGL graphics, so it might not be a reliable option.
- OpenGL ES API level: Select the maximum version of OpenGL ES to use in the emulator.
- Autoselect: Let the emulator choose the OpenGL ES version based on the host and guest support.
- Renderer maximum (up to OpenGL ES 3.1): Attempt to use the maximum version of OpenGL ES.
- Compatibility (OpenGL ES 1.1/2.0): Use the version of OpenGL ES that is compatible with most environments.

Help > Keyboard Shortcuts	This pane provides a complete list of keyboard shortcuts for the emulator. To open this pane while working in the emulator, press F1 (Command+/ on Mac).
Help > Emulator Help	To go to the online documentation for the emulator, click Documentation. To file a bug against the emulator, click Send feedback.
Help > About	See which adb port the emulator uses, as well as the Android and emulator version numbers. Compare the latest available emulator version with your version to determine if you have the latest software installed. The emulator serial number is emulator-adb_port, which you can specify as an adb command line option, for example.

Wi-Fi

При використанні AVD з рівнем API 25 або вище емулятор забезпечує імітацію точки доступу Wi-Fi («AndroidWifi»), і Android автоматично підключається до нього.

Ви можете відключити Wi-Fi в емуляторі, запустивши емулятор з параметрами командного рядка: *-feature -WiFi.*

Обмеження

Android Emulator не містить віртуального обладнання для таких дій:

- Bluetooth;
- NFC;

- SD-карта вставки/вилучення;
- навушники підключеного пристрою;
- USB.

Емулятор для годинника Wear OS не забезпечує підтримку кнопки Останні програми, D-Pad і датчика відбитків пальців.

6.2. Тестування за допомогою Web-сервісів

Також існує можливість використовувати сервіси для віддаленого доступу та управління мобільними пристроями. Вони являють собою веб-інтерфейс до «ферм» реальних пристроїв, з якими можна взаємодіяти як в ручному, так і в автоматичному режимі.

Переваги Web-сервісів :

- немає необхідності купувати пристрої;
- миттєвий доступ;
- засоби автоматизації тестування;
- можливість підключення до різних мобільних операторів (навіть в інших країнах).

Недоліки Web-сервісів:

- не всі пристрої є в наявності;
- популярні пристрої можуть бути зайняті;
- не всі дії користувача можуть моделюватися;
- ціна використання може бути досить високою.

Приклади таких сервісів показано на рис. 6.5, 6.6.

https://portal.deviceanywhere.com https://www.perfecto.io/

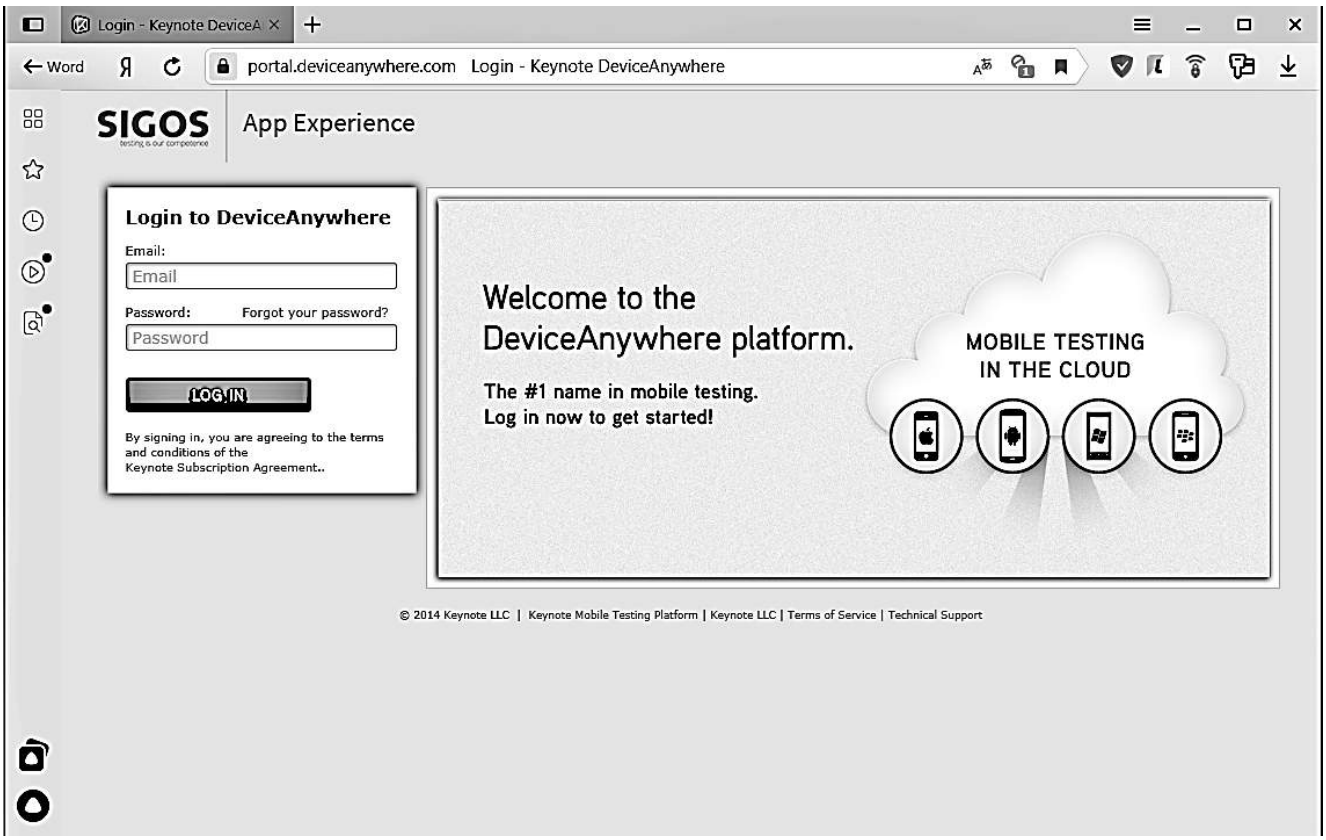


Рис. 6.5. Інтерфейс DeviceAnyWhere

Рис. 6.6. Інтерфейс Perfecto

Для полегшення поширення тестових версій програм використовують додаткові сервіси, тому що більшість виробників операційних систем для мобільних пристроїв не підтримують тестувальників у цьому. Виняток - Google, який дозволяє збирати статистику про використання додатка, його завантаження, «зльоти», а також поширювати додаток всередині команди з використанням сервісів Google.

Прикладом стороннього сервісу є TestFlight. За допомогою цього сервісу можна розміщувати додатки на сервері, відправляти вибраним тестувальникам повідомлення з посиланням на нову версію для установлення, за допомогою SDK - отримувати відгуки, інформації про місця дефектів, організовувати опитування тестерів (рис. 6.7).

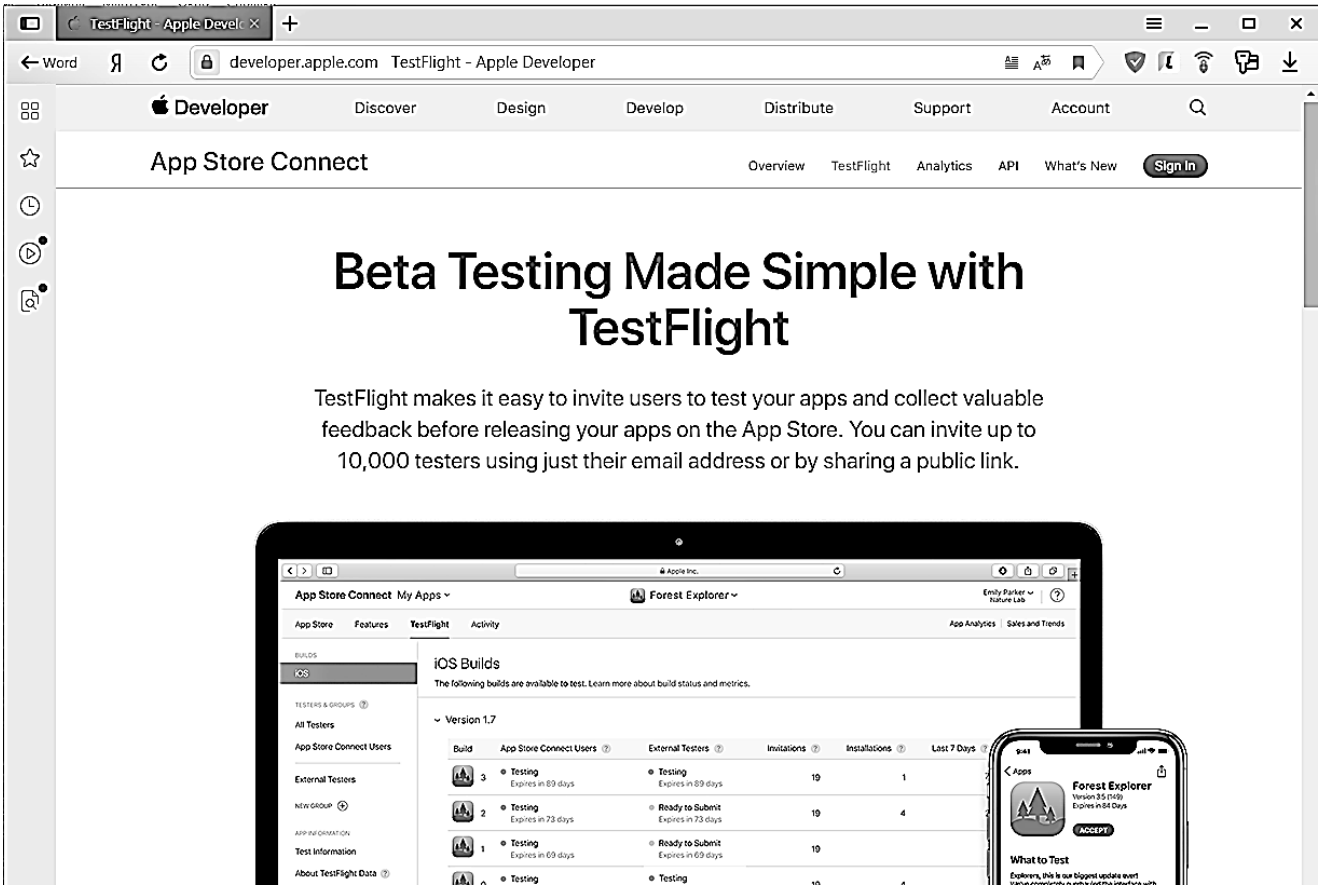


Рис. 6.7. Інтерфейс TestFlight

7. ІНСТРУМЕНТИ ДЛЯ ТЕСТУВАННЯ МОБІЛЬНИХ ДОДАТКІВ

Google Play Developer Console

Цей сервіс дозволяє розмістити додаток на мобільний маркет, але при цьому обмежити доступ до додатка тільки певною групою користувачів. Передбачені стадії альфа-тестування, бета-тестування і власне реліз, коли посилання стає доступним усім. Тестувальників

необхідно додати в спеціальну групу на Google+, яку задає розробник. Після цього йому будуть доступні посилання у маркеті. Для інших користувачів цей додаток недоступний ані через пошук, ані за прямим посиланням. Такий підхід дозволяє легко керувати тестованими версіями, збирати статистику на різних пристроях і т. д. (рис. 7.1).

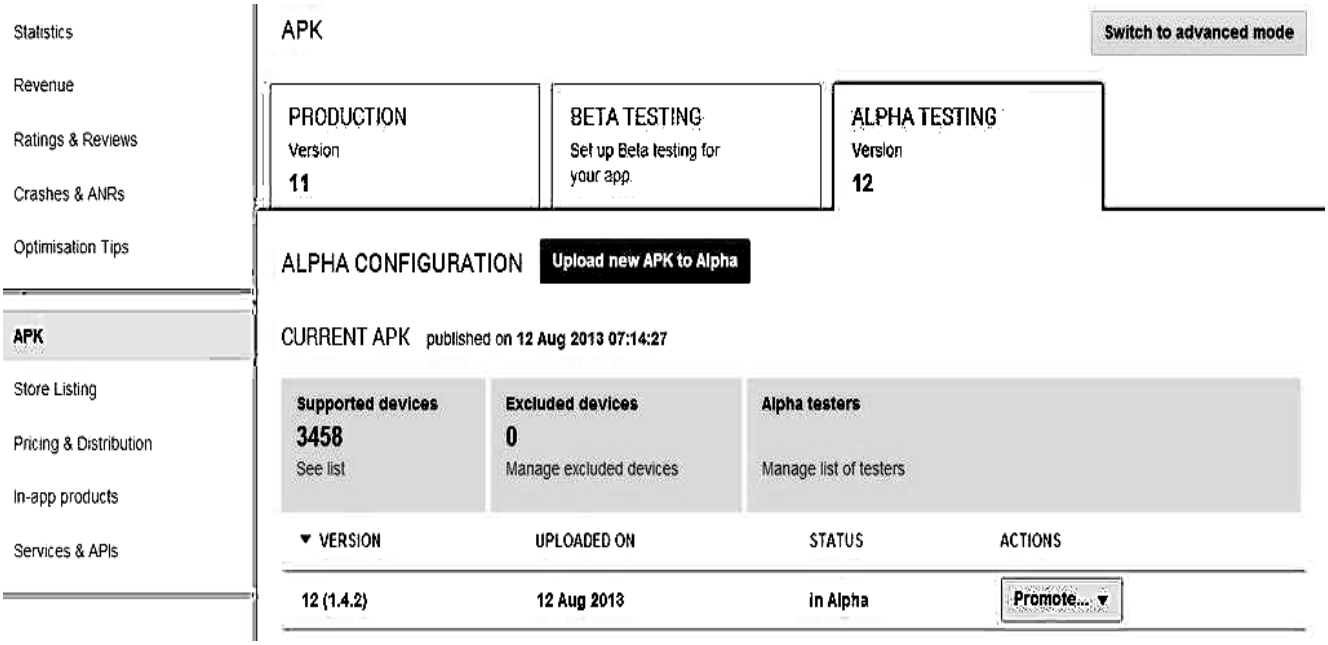


Рис. 7.1. Інтерфейс Google Play Developer Console

З урахуванням великої кількості різноманітних пристроїв, дозволів екрана, регресійних тестів і т. д. кількість повторюваних тестів дуже велика. Тому гостро стоїть питання автоматизації тестування.

7.1. Автоматизоване відтворення скриптових тестів

При виборі будь-якого інструменту слід брати до уваги принцип його роботи. Найбільш поширені два варіанти:

- Відтворення тесту відбувається за зверненням до екрана, без аналізу самого екрана і елементів інтерфейсу. Зазвичай таке відтворення здійснюється через координати жестів на екрані. Головний плюс - зазвичай немає необхідності модифікувати додаток. Головний мінус - залежність тестів від розміру екрана, орієнтації пристрою, дизайну програми.
- Відтворення тесту за допомогою звернення до інтерфейсних елементів програми. У тесті вказані мітки для форм, кнопок, текстбоксів та іншої візуальної «начинки». Головний плюс - навіть істотні зміни в інтерфейсі додатка навряд чи вплинуть на роботу тесту. Головний мінус - доведеться просити розробників збирати тестувальникам версії додатків з додатковими бібліотеками.

UIAutomation - стандартне рішення від Apple, яке дозволяє виконувати написані на JavaScript тестові сценарії як в емуляторі, так і на пристрої. UIAutomation входить до складу Instruments. Компілювати додаткові бібліотеки не потрібно. Починаючи з Xcode 4.3 з'явилася можливість запису тестів через рекодер.

Robotium - мабуть, найвідоміший на цей момент інструмент для автоматизації тестування додатків Android. "It's like Selenium, but for Android" стверджують розробники. Тести пишуть на Java (є сторонні рішення, що дозволяють писати їх, наприклад, на Python). Можливості запускати тести на пристрої немає, тільки - в емуляторі. Необхідно додавати бібліотеку до збірки додатків.

MonkeyRunner - поставляється в складі Android SDK, дозволяє виконувати функціональне тестування програми під Android, надаючи API для керування пристроєм. MonkeyRunner є більш низькорівневим порівняно з Robotium і не потребує початкового коду програми. Тести пишуться на Python або за допомогою рекордера, виконуються як в емуляторі, так і на реальних пристроях, підключених до комп'ютера. Великий мінус цього рішення у тому, що жести записуються в координатах, перевірку результатів здійснюють тільки шляхом порівняння скриншотів, що дуже ускладнює використання одного скрипта для тестування на декількох пристроях, а так само робить скрипти такими, що не відповідають регресійному тестуванню у разі зміни GUI додатка.

TestStudio - безкоштовний додаток для автоматичного тестування на платформі iOS. Базується на зверненні до компонентів додатка, а не на скриншотах. У ньому є як рекодер, так і можливість писати і редагувати тести вручну. Дозволяє тестувати web і native компоненти. Цікава особливість: можна записувати і відтворювати тести на пристрої без підключення до комп'ютера.

AppThwack - цікавий сервіс для тестування на Android-пристроях (підтримка iOS очікується незабаром). Ви завантажуєте свій додаток на ресурс, він встановлюється на реальні пристрої (асортимент налічує близько сотні пристроїв) і піддається «дослідженню» - запуск, заміри використовуваної пам'яті і завантаження процесорів, виявлення помилок і проблем, навантаження невеликим monkey-тестом. За результатами дослідження створюється звіт з скриншотом. Має тріал на тиждень, ціна - від 29 $ на місяць за тестування на 10 найбільш поширених пристроях.

JamoSolution - одна з найбільш багатообіцяючих платформ, на якій зараз розробляється кілька інструментів (наприклад, M-eux test і SeeTest). Вона дозволяє тестувати iOS, Android, Windows Phone та інші платформи. Підтримується запис тестів (record & play). Працює через установку на пристрої додатки-агенти, що вивільняє розробника від модифікування свого застосування. Має тріальну версію.

EggPlant - від студії TestPlant дозволяє запускати свій тестовий скрипт на безлічі пристроїв одночасно, визначаючи вихідні дані методом

розпізнавання картинки на екрані. Підтримує тестування на пристроях Android і iOS і їх емуляторах, а так само на емуляторі Windows Phone. Додаток розроблений під Windows, Linux, Mac. Має тріальну версію.

Squish - платний (2400 $) засіб для автоматичного тестування Qt, Web, Java, iOS та інших додатків. Підтримує запис тестів, сприймає скрипти на Javascript, Python, Perl або TCL. Має 30-денну тріальну версію.

Sikuli - open source інструмент для автоматизації тестування GUI Java-додатків (у тому числі і Android). Відкрите кросплатформене візуальне середовище створення сценаріїв-скриптів, яке орієнтоване на програмування графічного інтерфейсу за допомогою зображень (скриншотів). Особливість - скрипт, що задає послідовність дій, дозволяє використовувати скриншоти - щоб дати команду натиснути кнопку, для цього досить підставити в скрипт скриншот цієї кнопки. Підтримує написання скриптів на Java і Python.

MonkeyTalk - безкоштовний інструмент для тестування Android і iOS-додатків. Має власну потужну скриптову мову (можна писати скрипти і на Javascript), дозволяє створювати і зберігати тестові проекти (тест-кейси, тест-сьюти). Так само існує інтеграція з Eclipse, є рекодер. Потребує вставки своєї бібліотеки в додаток.

Robot Framework - це open-source фреймворк для автоматизації приймального тестування і розроблення через приймальні випробування (ATDD), що має широкий функціонал. Підтримує додаткові бібліотеки (можна використовувати власні, написані на Python або Java) - саме за допомогою уже реалізованих бібліотек і з'являється можливість тестування додатків на Android і iOS.

Тестування навантаження

HP Virtual User Generator (HP VuGen) - служить для навантаження сервера мультиплікаційним вхідним трафіком. Тестувальник створює скрипт, запускає його на пристрої, HP VuGen перехоплює трафік й імітує запити до сервера з подібною інформацією від кількох тисяч або мільйонів пристроїв одночасно.

Neoload - дозволяє емулювати правдоподібні умови мережі (емуляція 3G, 3G+, H+, 4G LTE). Може налаштовуватися для емуляції навантаження з різноманітного списку пристроїв (є передустановлення для пристроїв iPhone5, Samsung Galaxy Tab II, Nokia Lumia 800, Blackberry Bold 9900 та інших), основний скрипт навантаження можна записати за допомогою будь-якого реального пристрою за допомогою Wi-Fi. Має безкоштовний місячний тріал, вартість найдешевшої ліцензії (на п'ять користувачів) - 1200 євро. Підтримує пристрої на базі iOS, Android, Windows Phone та інші.

Monkey-тестування

Monkey - є інструментом стрес-тестування для Android, що міститься в Android SDK. Генерує псевдовипадкові дії користувача. Дозволяє налаштувати «хаотичність», інтервал між подіями, їх тип і т. п. Модифікувати код програми не потрібно. Тестувати можна як на емуляторі, так і на підключеному пристрої.

Anteater - безкоштовний інструмент для monkey-тестування iOS-додатків. Має більш широкий функціонал, ніж Monkey для Android. Існує тільки під Mac.

Сервіси для бета-тестування

uTest - спільнота з 45 тисячами професійних тестувальників із 180 країн. Реальні користувачі протестують роботу додатка. Сервіс платний. (IOS, Android, Windows Phone).

The Beta Family - безкоштовний сервіс для тестування програми. Заводимо аккаунт, заливаємо бета-версію програми, розсилаємо запрошення на тестування, обробляємо результати тестування. Можна вибрати тип бета-тестувальників: private або public. Якщо public, то додаток зможуть тестувати всі бажаючі. Працює з iOS, Android, Windows Phone.

Zubhium - надає SDK для Android, за допомогою якого Ви в свій додаток убудовуєте код для автоматичного збору інформації про помилки. Розробник викладає бета-версію, запрошує тестувальників, отримує інформацію. Вартість передплати від 10 $/міс. Має місячну тріальну версію.

Збирачі статистики

Треба знати, як користувачі працюють з вашим додатком: які функції найбільш затребувані, які кнопки натискаються, які налаштування змінюються, які помилки трапляються, скільки часу користувач проводить у різних вікнах. Непогано також мати статистику щодо користувачів - яка версія ОС у їх девайсів, де вони географічно розташовані і т. д. Найпростіший шлях збору такої статистики - скористатися готовою системою збору аналітичної інформації. Наприклад:

- Flurry (Безкоштовна) (iOS, Android, Windows Phone);
- BugSence (Безкоштовна) (iOS, Android, Windows Phone);
- Apsalar (Безкоштовна) (iOS, Android);
- Google Analytics (Безкоштовна) (iOS, Android);
- Mixpanel (Платна) (iPhone, Android);
- Localytics (Платна) (iOS, Android, Windows Phone);

- Bango (Платна) (Android, Windows Phone).

У кожної системи є свої родзинки: оновлення статистики в реальному часі (Localytics), суперточна з відстеженням унікальних ID кожного користувача (Bango), наявність коштів для проведення опитування серед користувачів (Apsalar) і т. д. Природно, є і багато відмінностей: в інтерфейсі, засобах аналізу, в наявності додаткових API, у вартості, наборі підтримуваних платформ.

Інші корисні інструменти

Fake GPS - додаток для пристроїв Android, що дозволяє встановити довільні дані в модулі геолокації. Так само як в раніше згадуваному Android SDK, є функції, що полегшують тестування додатків під Android - консольні можливості установлення, видалення та запуску додатків, перегляд в реальному часі та вивантаження логів роботи пристрою у файл, перезавантаження додатка і т. д. Опис усіх цих можливостей легко знайти в Інтернеті.

Комплексні рішення

TestQuest Pro - інструмент для повного автоматичного тестування, розроблений для компаній і підприємств. Підтримує функціональне, навантажувальне, регресивне тестування, тестування якості та взаємодії.

Experitest.com - дозволяє проводити автоматичне, ручне (віддалено) і хмарне тестування на великому спектрі пристроїв (2500 $ у рік за автоматичне тестування, 4000 $ у рік - за ручне).

Варто відзначити існування компаній, що спеціалізуються на тестуванні, в тому числі й на тестуванні додатків на мобільних пристроях. наприклад, **Qulix QA** виробляє всебічне тестове покриття - верифікація роботи програми щодо ОС, платформ, мов та ін.; проходження сертифікації для електроного підпису продуктів і попадання в маркет платформи; тестування додатків на реальних мобільних пристроях.

8. ПРИКЛАД ЧЕК ЛИСТА ДЛЯ МОБІЛЬНОГО ТЕСТУВАННЯ

Цей перелік спеціально розроблений, щоб перевірити характеристики мобільного додатка. Очевидно, що він перевіряє тільки загальні характеристики. Для цього окремого тестового підходу і тестовий скрипт повинен бути створений. Те ж саме і для тестування продуктивності, юзабіліті-тестування, тестування безпеки та інших заходів, необхідних для тестування вашого конкретного додатка.

Контрольний список розділений на п'ять різних областей:

- Специфічні характеристики пристрою. Це характеристики, які пов'язані з пристроєм, на якому встановлена програма.
- Мережа спеціальних перевірок.
- Перевірки функціональності додатка.
- Перевірки інтерфейсу користувача.
- Спеціальні перевірки.

Перевірки мають бути виконані в порядку, в якому вони розташовані. Оскільки 95 % IT-проектів в Україні виконуються англійскою мовою, то цей чек лист наводимо мовою оригиналу.

Device specific checks

#	Description	OK/NOK?	Remarks
1.1	Can the app be installed on the device?	N/A	
1.2	Does the app behave as designed/desired if there is an incoming call?	N/A	
1.3	Does the app behave as designed/desired if there is an incoming SMS?	N/A	
1.4	Does the app behave as designed/desired if the charger is connected?	N/A	
1.5	Does the app behave as designed/desired if the charger is disconnected?	N/A	
1.6	Does the app behave as designed/desired if the device goes to sleeping mode	N/A	
1.7	Does the app behave as designed/desired if the device resumes from sleeping mode	N/A	
1.8	Does the app behave as designed/desired if the device resumes from lock screen?	N/A	
1.9	Does the app behave as designed/desired if the device is tilted?	N/A	
1.10	Does the app behave as	N/A	

	designed/desired if the device is shaken?		
1.11	Does the app behave as designed/desired if a local message is coming from another app (think of: calendar reminders, to-do task etc.)	N/A	
1.12	Does the app behave as designed/desired if a push message is coming from another app (think of: twitter mentions, whatsapp message, wordfeud invitation, etc)	N/A	
1.13	Does the app interact with the GPS sensor correctly (switch on/off, retrieve GPS data)?	N/A	
1.14	Is the functionality of all the buttons or keys on the device defined for this app?	N/A	
1.15	Verify that buttons or keys which have no defined function have no unexpected behaviour on the app when activating	N/A	
1.16	In case there's a true "back" button available on the device does the "back" button take the user to the previous screen?	N/A	
1.17	In case there's a true "menu" button available on the device, does the menu button show the app's menu?	N/A	
1.18	In case there's a true "home" button available on the device, does the home button get the user back to the home screen of the device?	N/A	
1.19	In case there's a true "search" button available on the device, does this get the user to some form of search within the app?	N/A	
1.20	Does the app behave as designed/desired if the "Battery low" message is pushed	N/A	

1.21	Does the app behave as designed/desired if the sound on the device is turned off?	N/A	
1.22	Does the app behave as designed/desired if the device is in airplane mode?	N/A	
1.23	Can the app be de-installed from the device?	N/A	
1.24	Does the application function as expected after re-installation?	N/A	
1.25	Can the app be found in the app store? (Check after go-live)	N/A	
1.26	Can the app switch to different apps on the device through multitasking as designed/desired?	N/A	
1.27	Are all touch screen positions (buttons) working when a screen protector is used	N/A	

Network specific Checks

#	Description	OK/NOK?	Remarks
2.1	Does the app behave according to specification if connected to the internet through Wi-Fi?	N/A	
2.2	Does the app behave according to specification if connected to the internet through 3G?	N/A	
2.3	Does the app behave according to specification if connected to the internet through 2G?	N/A	
2.4	Does the app behave according to specification of the app is out of network reach?	N/A	
2.5	Does the app resume working when it gets back into network reach from outside reach of the network?	N/A	
2.6	Update transactions are processed correctly after re-establishing	N/A	

	connection		
2.7	Does the app still work correctly when tethering or otherwise connected to another device	N/A	
2.8	What happens if the app switches between networks (Wi-Fi, 3G, 2G)	N/A	
2.9	Does the app use standard network ports (Mail: 25, 143, 465, 993 or 995 HTTP: 80 or 443 SFTP: 22) to connect to remote services, as some providers block certain ports	N/A	

App specific Checks

#	Description	OK/NOK?	Remarks
3.1	Has the app been tested on different type of devices and different versions of OS?	N/A	
3.2	Stability check: if the app has a list (for instance of pictures) in it, try scrolling through it at high speed	N/A	
3.3	Stability check: if the app has a list (for instance of pictures) in it, try scrolling to before the first picture or behind the last picture	N/A	
3.4	Is downloading of the app prevented in case it's bigger than the OS allows downloading when connected to cellular networks	N/A	
3.5	Integration: does the app connect correctly to the different social networks (LinkedIn, twitter, facebook, etc)	N/A	
3.6	The app does not interfere with other apps when in background/multitasking mode (using GPS, playing music, etc.)	N/A	
3.7	Can the user print from the app (if applicable)	N/A	
3.8	The search option in the app displays relevant results	N/A	

3.9	Verify most common gestures used to control the app	N/A	
3.10	What happens if you select different options at the same time (undesired multitouch, for example – select two contacts from the phone book at the same time)	N/A	
3.11	App name should be self explanatory	N/A	
3.12	Does the app limit or clean the amount of cached data	N/A	
3.13	Reloading of data from remote service has been properly designed to prevent performance issues at server-side. (manual reloading of data can reduce the amount of server calls)	N/A	
3.14	Does the app go to sleep mode when running in the background (prevent battery drain)	N/A	

User interface checks

#	Description	OK/NOK?	Remarks
4.1	To keep controls as unobtrusive as possible for instance by fading them out if they are not used for a while	N/A	
4.1	Make it possible for users to go back to a previous screen for instance by adding a back or cancel button	N/A	
4.2	The main function of the app should be apparent immediately. It should speak for itself	N/A	
4.3	Use at most one action on the screen that is highlighted as the most likely for the user. (Example: in iOS a blue button represents the default or most likely action)	N/A	
4.4	Minimize user actions by using a picker or a table view where users can select a certain choice over a	N/A	

	data entry field where users have to type a choice		
4.5	In an app, the user should not be able to store files locally, outside the app sandbox	N/A	
4.6	In an app, the user should not be exposed to the permissions of a specific file	N/A	
4.7	If there is a long list of data to scroll trough, provide a search option above the list	N/A	
4.8	If performance is slow, indicate a progress status icon (“Loading…”), preferably with specific message	N/A	
4.9	In case of ‘live’ filtering of data while the user enters his search query, verify the performance	N/A	
4.10	The appearance of buttons that perform standard actions are not altered in the app (for instance: refresh, organize, trash, Reply, back, etc.)	N/A	
4.11	Do not use standard buttons for other functions then that they are normally used for	N/A	
4.12	The app should respond to all changes in device orientation, as per the design	N/A	
4.13	Tapable elements should be about 7x7 mm in size, using the pixel density of the target device you can calculate the amount of pixels (chapter documentation contains a link to different devices compared)	N/A	
4.14	Do not redefine gestures in your app that have a standard meaning (example: swiping from top to bottom enables the notification center)	N/A	
4.15	Requirement to login is delayed in the app as long as possible	N/A	
4.16	If the app is stopped at an unexpected time, user data should	N/A	

	be saved locally and available at start-up		
4.17	Users should be warned of the consequences of deleting a document	N/A	
4.18	Keyboard adjusts to expected input (for instance numbers/letters when expected)	N/A	
4.19	Are inactive buttons clearly distinguished from active buttons?	N/A	

Store specific Checks

#	Description	OK/NOK?	Remarks
5.1	The app can't use any "non-public API's". This means that you can't use some functions that the distributing platform uses for its own apps. (This can generally be checked best by some sort of automated tool, like http://www.chimpstudios.com/appscanner/)	N/A	
5.2	The app can't reprogram controls of the device that are not intended for that use. (For instance: using the volume button as a shutter for the camera)	N/A	
5.3	The app should not access information on the device outside the app without the user's permission (for instance, copying the address book or getting information from other apps)	N/A	
5.4	The app should not access or write files outside the "Bundle" and "Documents" directory. (because the app can't read or write data outside the designated container area)	N/A	
5.5	The app cannot download code to be installed without the users consent	N/A	

5.6	The app can only get new functionality by way of an upgrade through the app store	N/A	
5.7	After download, an app should remain working. An app cannot turn off after a few days	N/A	
5.8	An app can't be a "trail", "beta", "demo" or "test" version	N/A	
5.9	Apple product names should be spelled correctly in the app. (For instance: IPhonez is wrong)	N/A	
5.10	If the app uses the web, it is **not** done using third party (i.e. non-Apple) browsers	N/A	
5.11	You cannot mention other app platforms in your app (for instance: "Also available on android!")	N/A	
5.12	An app cannot use old interfaces, like for instance the iPod click wheel	N/A	
5.13	Multitasking functionality of the app can only be used for its intended purposes, i.e. VoIP, Audio playback, location, task completion, local notifications, etc. This means that generally an app can't run in the background but has to be closed off if it's not used any more	N/A	
5.14	The app must have some functionality. For instance, it can't be just a title page leading to some text. It can't be just a song, movie or book as there are different platforms for that	N/A	
5.15	Functionality should be in sync with functionality described in store	N/A	
5.16	In general, the app has to be *decent*. So no explicit material in the sense of sex, violence, drugs, alcohol or tobacco. It cannot	N/A	

	address a specific ethnic or religious group in a derogatory way		
5.17	The app has to be *honest*. This means that the description of the app has to be correct, and all functionality has to work as described. If an app gives diagnostic information, it has to be reliable. This also means that the genre and category in the description must be appropriate. The app icons should be consistent and appropriate	N/A	
5.18	An app can't restrict the users of the app for instance by location or carrier	N/A	
5.19	An app cannot send spam or introduce viruses, or use other apple platforms like Game Center and Push Notifications to do so	N/A	
5.20	The app should aim at backing up a minimum of information on iCloud. The information in iCloud should be just the user generated information. Information that can be recreated or downloaded should not be backed up	N/A	
5.21	An app cannot use location services of the device without asking permission	N/A	
5.22	All url's in the app code should be fully functional	N/A	
5.23	The app can't use the user's location without permission	N/A	
5.24	The location services cannot be used to autonomously control of vehicles or for emergency services	N/A	
5.25	An app cannot use push notifications without user consent	N/A	
5.26	Push notifications have to be send using the Apple Push Notification	N/A	

	(APN) API. This has to be done using an APN ID		
5.27	Push notification can’t send personal information	N/A	
5.28	The App may not distribute any private information of users (like Player ID) through the game center	N/A	
5.29	Ad banners must be hidden when there are no ads available	N/A	
5.30	The app must respect copyright of apple and other parties	N/A	
5.31	The in app purchase mechanism cannot be used to purchase goods and services used outside the app	N/A	
5.32	The in app purchase mechanism cannot be used to collect money for charities. This has to be done through SMS	N/A	
5.33	The in app purchase mechanism cannot be used to buy a raffle or lottery ticket directly from the app	N/A	
5.34	Apps that encourage the users to use the device in a way that may damage the device will be rejected	N/A	
5.35	An app cannot require user’s personal information (for instance email address) in order for it to function	N/A	

БІБЛІОГРАФІЧНИЙ СПИСОК

1. Канер, С. Тестирование программного обеспечения. Фундаментальные концепции тестирования бизнес-приложений / С. Канер, Дж. Фолк, Е. Нгуен. Киев : – Диасофт, 2001. – 544 с.

2. Савин, Р. Тестирование Дот Ком, или пособие по жестокому обращению с багами в интернет-стартапах / Р. Савин. – М. : Дело, 2007. – 312 с.

3. Губка, С. А. Основы тестирования информационных управляющих систем: учеб. пособие / С. А. Губка, А. С. Губка, П. Е. Ельцов. – Харьков : Нац. аэрокосм. ун-т «Харьков. авиац. ин-т», 2016. – 67 с.

4. Про Тестинг - Тестирование Программного Обеспечения [Электронный ресурс]. – Режим доступа: www.protesting.ru - 10.07.2020.

5. Тестирование и качество ПО [Электронный ресурс]. – Режим доступа: https://software-testing.ru/ - 10.07.2020.

6. Руководство по разработке приложений под Android/ [Электронный ресурс]. – Режим доступа: https://developer.android.com/design - 10.07.2020.

Навчальне видання

Губка Олексій Сергійович
Губка Сергій Олексійович

ОСОБЛИВОСТІ ТЕСТУВАННЯ МОБІЛЬНИХ ДОДАТКІВ

Редактор С. П. Гевло

Зв. план, 2020
Підписано до друку 27.07.2020
Формат 60х84 1/16. Папір офс. № 2. Офс. друк
Ум. друк. арк. 4,44. Обл.-вид. арк. 5. Наклад 50 пр.
Замовлення 194. Ціна вільна

Видавець і виготовлювач
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»
61070, Харків-70, вул. Чкалова, 17
http:// www.khai.edu
Видавничий центр «ХАІ»
61070, Харків-70, вул. Чкалова, 17
izdat@khai.edu

Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру видавців, виготовлювачів і розповсюджувачів
видавничої продукції сер. ДК № 391 від 30.03.2001