ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення автоматизованих систем

Пояснювальна записка

до магістерської роботи
на ступінь вищої освіти магістр

на тему: **“ РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ НА БАЗІ ANDROID З РОЗШИРЕНИМИ МОЖЛИВОСТЯМИ ТА ПОКРАЩЕННОЮ АДАПТИВНІСТЮ”**

Виконав: студент 6 курсу, групи ІСДМ-61
спеціальність

126 Інформаційні системи та технології
(шифр і назва спеціальності)

Кашич Д.О.
(прізвище та ініціали)

Керівник Бондарчук А.П.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення автоматизованих систем
Ступінь вищої освіти магістр
Спеціальність 126 Інформаційні системи та технології.
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри ІПЗАС
_____ К.П.Сторчак
“____” _____2022 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Кашичу Дмитрію Олександровичу
(прізвище, ім’я, по батькові)

1.Тема роботи: «Розробка інформаційної системи на базі Android з розширеними можливостями та покращеною адаптивністю "
керівник роботи Бондарчук Андрій Петрович, доктор технічних наук, професор
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “ 03 ” 12 2022 року № 116

2. Строк подання студентом роботи 26.12.2022

3. Вихідні дані до роботи:
1. Android система.
2. Платформа мобільного додатку.
3. Класифікація Android платформи.
4. Архітектура Android платформи.
5. Середовище мобільного додатку.
6. Науково-технічна література.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити:
1. Аналіз розвитку Android систем.
2.Перспективи розвитку використання Android систем в інформаційних системах.
3. Практична реалізація інформаційної системи на базі Android.

5.Перелік графічного матеріалу (з точним зазначенням обов'язкових слайдів)

1. Мета та завдання.
2. Загальні положення Android систем.
3. Архітектура Android систем.
4. Перспективи використання Android систем в інформаційних системах.
5. Структура Android додатків.
6. Сучасний стан та перспективи застосування Android додатків.
7. Генерація стратегії системи.
8. Алгоритм для фільтрування проєктів, новин та подій.
9. Аналіз сучасних Android середовищ та платформ.
10. Випробування інформаційної системи.
11. Моделювання в середовищі Android Studio.

6. Дата видачі завдання 03.10.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1.	Ознайомлення з завданням, огляд та підбір технічної літератури для виконання поставлених задач		Виконано
2.	Аналіз Android середовищ та платформ		Виконано
3.	Дослідження перспективи використання Android систем в інформаційних системах		Виконано
4.	Практична реалізація інформаційної системи		Виконано
5.	Аналіз отриманих результатів дипломної роботи. Підсумки роботи		Виконано
6.	Розробка демонстраційного матеріалу		Виконано

Студент __

Керівник роботи _____

РЕФЕРАТ

Текстова частина магістерської роботи 62 сторінки, 30 рисунків, 1 таблиця.

Мета роботи – розробка інформаційної системи на базі платформи Android з розширеними можливостями та покращеною адаптивністю. Цілі розробки являють собою розширення можливостей інформаційної системи, а саме, фільтрації подій, новин, програма благодійності та покращення адаптивність на різних девайсах з ОС Android.

Розділ «Аналіз інформаційних систем на базі Android» описує загальні положення Android систем, платформи мобільних додатків, а також класифікацію, архітектуру та характеристику платформ.

Розділ «Структура. Android додатків» розкриває в собі структуру додатків під платформу Android, компоненти Android додатків, яка активувати компоненти та маніфест файли. Детально розкриває використання ресурсів та їх керування в додатках Android.

Розділ «Практична реалізація розширених можливостей» містить в собі алгоритм для фільтрування проєктів, новин та подій, детальну генерацію стратегії системи та випробування інформаційної системи.

Об'єкт дослідження – Android системи

Предмет дослідження – алгоритм фільтрування подій, новин та проєктів

Дипломний проєкт присвячений розробці інформаційної системи на базі Android, основною метою є розширення можливостей та покращена адаптивність.

Галузь використання – інформаційні системи.

Ключеві слова: ANDROID СИСТЕМА, KOTLIN, ІНФОРМАЦІЙНА СИСТЕМА, АДАПТИВНІСТЬ, АЛГОРИТМ ФІЛЬТРУВАННЯ ПРОЄКТІВ, MVVM, ANDROID STUDIO, ДОДАТОК ANDROID.

ЗМІСТ

Стор.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

SDK – software development kit

CSS - Cascading Style Sheets

Java ME – Java Micro Edition

CLDC – Connected, Limited Device Configuration

CDC – Connected Device Configuration

СУБД – системами управління бази даних

MIDP – Mobile Information Device Profile

CDMA – Code Division Multiple Access

GSM – Groupe Special Mobile

NFC – Near Field Communication

RIM – Research In Motion Limited

ОС – операційна система

ID (Identifier) – ідентифікатор

PWA – progressive web app

IDE – Integrated development environment

ВСТУП

Розробка мобільних додатків — це набір процесів і процедур, пов’язаних із написанням програмного забезпечення для невеликих бездротових комп’ютерних пристроїв, таких як смартфони та інші кишенькові пристрої.

Як і розробка веб-додатків, розробка мобільних додатків бере свій початок у більш традиційній розробці програмного забезпечення. Проте одна критична відмінність полягає в тому, що мобільні програми часто пишуться спеціально для використання переваг унікальних функцій конкретного мобільного пристрою.

Сьогодні двома найвідомішими мобільними платформами є iOS від Apple і Android від Google. Телефони та планшети від Apple постачаються з попередньо встановленими основними програмами, включаючи повноцінний веб-браузер і Apple App Store. Пристрої Android також постачаються з попередньо встановленими подібними програмами, і ви можете встановити більше за допомогою Google Play Store.

Розділ «Аналіз інформаційних систем на базі Android» описує загальні положення Android систем, платформи мобільних додатків, а також класифікацію, архітектуру та характеристику платформ.

Розділ «Структура. Android додатків» розкриває в собі структуру додатків під платформу Android, компоненти Android додатків, яка активувати компоненти та маніфест файли. Детально розкриває використання ресурсів та їх керування в додатках Android.

Розділ «Практична реалізація розширених можливостей» містить в собі алгоритм для фільтрування проєктів, новин та подій, детальну генерацію стратегії системи та випробування інформаційної системи.

1 АНАЛІЗ ІНФОРМАЦІЙНИХ СИСТЕМ НА БАЗІ ANDROID

1.1 Загальні положення Android систем

Система – це сукупність взаємозалежних елементів, об'єднаних до виконання певної функції, що має у своїй властивостями, не зводяться до сумі властивостей окремих елементів.

Основні ознаки систем:

1) Цілісність та ділимість;

2) Наявність стійких зв'язків;

3) Організація;

4) Емерджентність (наявність таких властивостей, які притаманні системі загалом, але не властиві жодному з її елементів окремо).

Система (область розробки програмного забезпечення) – це комплекс коштів, які забезпечують її діяльність у межах встановлених функцій.

Такими засобами можуть бути:

- лінгвістичні;
- організаційні;
- математичні;
- технічні;
- програмні.

Мала система – це система, котра має обмеження за обсягом використання комплексу коштів.

Наприклад:

- мінімальний набір елементів, які забезпечують заявлені функції;
- обмеження за габаритами (обсягом);
- обмеження щодо необхідних ресурсів (технічні);
- обмеження щодо лінгвістичних засобів (набір мовних елементів);
- обмеження за програмними засобами (набір програмного забезпечення);
- мінімізація енергоспоживання.

Прикладом малих систем можуть бути:

- системи керування побутовими приладами;

- будь-які системи, що вбудовуються (спеціалізовані мікропроцесорні системи управління, які працюють, будучи вбудованими безпосередньо в пристрій, яким вони керують);

- системи на мобільних пристроях;

- системи, які висувають специфічні вимоги до енергоспоживання, габаритів, програмного забезпечення.

Мобільні пристрої – це багатофункціональні портативні (переносні пристрої, для використання яких необхідне підключення до мережних портів) або переносні (для їх використання застосовується бездротовий зв'язок) комп'ютери.

Програми, за допомогою яких керуються багатофункціональні мобільні пристрої, називаються мобільними додатками.

Мобільна платформа – це набір програмних продуктів та технологій, які спільно надають систему для розробки прикладного програмного забезпечення та забезпечують його функціонування на мобільному пристрої.

1.1.1 Огляд типів Android систем

Розробка мобільних додатків перш за все поділяють за певними критеріями, а, саме:

- цільова аудиторія;
- для яких пристроїв передбачено використання додатку;
- платформа, на якій додаток буде функціонувати.

Нативний додаток – це додаток, розроблений своєю (тотожною) мовою програмування для обраної платформи (наприклад: objective-c для ios, java для android, c# для windowsphone).

Причини використання:

1) працюють швидше та стабільніше, ніж додатки іншого типу;

2) дозволяють зняти функціональні обмеження браузерів щодо доступу до ресурсів пристрою.

Мобільна платформа надає інструментарій для розробників (SDK – software development kit), який дозволяє отримати доступ практично до всіх сервісів пристрою, а також призначений для спрощення процесу створення програм.

Веб-додаток – це програма, розроблена на HTML, JavaScript, CSS (Cascading Style Sheets — каскадні таблиці стилів) і вимагає для свого виконання встановленого та налаштованого браузера мобільного пристрою з виходом в Інтернет.

HTML використовується для розмітки елементів інтерфейсу. CSS описує візуальну складову та взаємне розташування віджетів та елементів управління.

Мова програмування JavaScript реалізує логіку програми.

Використовують з наступних причин:

1) можливість повного чи хоча б часткового повторного використання коду на різних платформах;

2) не потребує особливих вимог до графіки та використання апаратних засобів пристрою;

3) є величезний вибір інструментів, фреймворків, які прискорюють та спрощують процес розробки;

4) існування версії веб-сайту для настільного комп'ютера і є необхідність отримання доступу через мобільний пристрій.

Гібридна програма – це програма, в якій частково використовується нативна функціональність, а частково можливості веб-додатків.

(від нативних програм - можливість часткового доступу до ресурсів пристрою; від веб-додатків -підтримка HTML і робота в браузері).

Загальні положення використання:

1) можна поширити його одразу на безліч платформ;

2) загальна продуктивність і відгук інтерфейсу не є вирішальними;

3) можливість поширення (публікації) як готового продукту чи тимчасового замінника до виходу нативного додатка (запустити процес маркетингу).

1.2 Платформи мобільних додатків

Платформи мобільних пристроїв можна розділити на:

- платформи, що використовуються на мобільних пристроях різних виробників;
- платформи, орієнтовані використання лише у пристрої конкретного виробника.

Існують платформи, що підтримують мобільні пристрої різних додатків. Серед них наступні:

- Java ME;
- Android;
- BREW;
- Windows Phone.

А також існують закриті платформи, такі як:

- Symbian;
- Windows Mobile;
- Bada;
- MeeGo
- Maemo;
- Palm.

Серед мобільних платформ, є ті. що підтримують девайст конкретних виробників:

- BlackBerry;
- iOS.

BlackBerry - розроблена компанією Research In Motion для смартфонів та планшетів BlackBerry.

iOS - розроблена компанією Apple спочатку - для iPhone та iPod touch, пізніше для Apple TV та iPad.

1.2.1 Класифікація, архітектура та характеристика платформ

Основними поняттями для платформ є класифікація, архітектура даної платформи та характеристика.

Однією з таких є платформа JAVA ME.

Java Micro Edition – призначена для пристроїв з обмеженими ресурсами (наприклад: стільникових телефонів, персональних кишенькових комп'ютерів, ресиверів цифрового телебачення, програвачів дисків Blu-ray).

Відмінними рисами пристроїв з обмеженими ресурсами є:

- обмежена обчислювальна потужність;
- обмежений обсяг пам'яті;
- мінімальний розмір дисплея;
- живлення від портативної батареї;
- низькошвидкісні та недостатньо надійні комунікаційні можливості.

Дві конфігурації, які представляють дві категорії портативних пристроїв:

1) особисті мобільні пристрої, що не стаціонарно підключаються — підтримуються конфігурацією Connected, Limited Device Configuration (CLDC, конфігурація пристроїв, що підключаються з обмеженнями);
2) Постійно з'єднані мережні пристрої, які підтримуються конфігурацією Connected Device Configuration (CDC, конфігурація для підключених пристроїв).

Рівні для платформи Java Micro Edition, з верхнього до нижнього йдуть наступним чином:

- Java-додаток;
- профіль;
- конфігурація (бібліотека та/або Java віртуальна машина);
- операційна система девайсу;
- апаратне забезпечення девайсу.

Профіль містить вимоги до апаратної частини пристрою, а також мінімальний набір API.

Конфігурацією CLDC підтримується профіль Mobile Information Device Profile (MIDP), інколи називають мідлетом, який призначений для мобільних пристроїв (телефони, двосторонні пейджери тощо) з такими характеристиками:

- розмір екрана приблизно (як мінімум) 96x54 пікселів;
- глибина екрану 1 біт;
- клавіатура для роботи однією або двома руками, пристрій введення з сенсорного екрана;
- 128 Кб енергонезалежної пам'яті для MIDP-компонентів;
- 8 Кб енергонезалежної пам'яті для даних постійного зберігання;
- 32 Кб енергозалежної оперативної пам'яті для динамічної пам'яті;
- двосторонній бездротовий зв'язок.

Наступна серед платформ є саме Android. Вона працює на базі ядра Linux, адаптованого для мобільних пристроїв.

З'явилася у жовтні 2008 року. Розробник – Android Inc. Власник – Google.

Відмінності від попередньої платформи Java ME:

- підвищення ефективності управління живленням;
- підвищення безпеки;
- підтримку нового обладнання;
- покращення системи флеш-пам'яті;
- покращення повідомлень про помилки.

Платформа орієнтована за наступними параметрами:

- мова програмування Java (крос-платформність);
- відкритість (впровадження інновацій);
- різноманітний набір вбудованих інструментів для розробки (наприклад: керування живленням, інтерфейс, програмування на основі розташування, локальне сховище та інтеграція хмарних служб).

Архітектура містить в собі додаток, систему додатку, бібліотеки, середовище виконання, і це все прикріплено ядром Linux.

Також виділяють платформу BREW.

Binary Runtime Environment for Wireless (двійкове середовище виконання для бездротових пристроїв) - програмна платформа, що дозволяє порівняно швидко завантажувати програми та працювати з ними.

Розробник платформи – американська компанія Qualcomm.

(Спочатку платформа була призначена для підтримки технології CDMA (Code Division Multiple Access) - технологія множинного доступу з кодовим поділом каналів, але може використовуватися і для GSM (Groupe Special Mobile, перейменовану в Global System for Mobile Communications) - глобальний цифровий стандарт для мобільної стільникової зв'язку).

BREW орієнтована мови програмування C і C++, яка дозволяє розробляти програми на Java, якщо попередньо встановлена Java ME (у даному випадку буде втрата швидкості виконання).

Програми для платформи BREW попередньо сертифікуються (список змінюється відповідно до сертифікації нових платформ):

- фірма Qualcomm;
- оператором.

При цьому оператору надається можливість вилучити або повернути на доопрацювання будь-який додаток, який отримав негативну оцінку клієнтів.

Компанія Qualcomm пропонує для вільного поширення BREW SDK - інструментарію для розробки на платформі Microsoft Windows.

До складу входять:

- документація;
- заголовні файли;
- симулятор;
- приклади додатків з вихідним кодом;
- пакет допоміжних утиліт.

За своєю структурою BREW схожа з віртуальною машиною Java. API платформи надає можливість роботи з: телефонією,

- SMS/MMS,

- адресною книгою,

- записом та відтворенням аудіо/відео/фото інформації,
- мережевими технологіями,
- створення інтерфейсів користувача, - криптографією,
- базами даних тощо.

Основні бібліотеки, які надає компанія Qualcomm:

1) BUIW — бібліотека віджетів;
2) OpenGL ES - 3D графіка;
3) SQL - робота з базами даних;
4) Font Extensions - шрифти.

Платформа Windows Phone є наслідником платформи Windows Mobile та представлена світові з жовтня 2010 року. В собі містить наступні основні властивості:

- створення умов, що полегшують портування програмного забезпечення між планшетами, смартфонами та персональними комп'ютерами;
- браузер із функцією виявлення шкідливих сайтів;
- Підтримку багатоядерних процесорів;
- технологію обміну даними NFC (Near Field Communication – технологія бездротового зв'язку);
- архітектуру драйверів, що допускає і драйвери режиму користувача, і драйвери режиму ядра.

Архітектура даної платформи є таким, що в основі мова програмування C#, платформа для запуску Silverlight/XNA, сам додаток на C++, Win 32 API, Windows Runtime, DirectX 11.1, Ядро ОС (Kernel). Дизайн інтерфейсу в стилі "Метро", де елементи є плитки і є посиланнями на додатки, різні функції або індивідуальні об'єкти.

Також вище згадана платформа BlackBerry розроблена компанією Research In Motion Limited (RIM).

В основному була націлена на корпоративного користувача і тому найбезпечнішою (використовується концепція мікроядер для реалізації різних функцій безпеки).

Особливість – використання спеціального сервера (Blackberry Enterprise Server, BES) та шифрування за стандартом AES (прийнятий у США) для захисту повідомлень від перехоплення.

Принцип дії поштових мереж BlackBerry:

- індивідуальний ідентифікаційний номер (PIN);
- передача повідомлень PIN;
- система «peer-to-peer».

Остання версія – BlackBerry 10. Структура робочого столу схожа на робочі столи систем iOS та Android. Клавіатура - віртуальна QWERTY-клавіатура з особливим дизайном (підказки, що спливають слова). Відсутня кнопка увімкнення (достатньо провести пальцем по екрану). Інші програми встановлюються з магазину програм BlackBerry World. Мова розробки - Java, проте підтримує і додатки, написані на C/C++ та HTML5.

Архітектура BlackBerry має наступний вигляд:

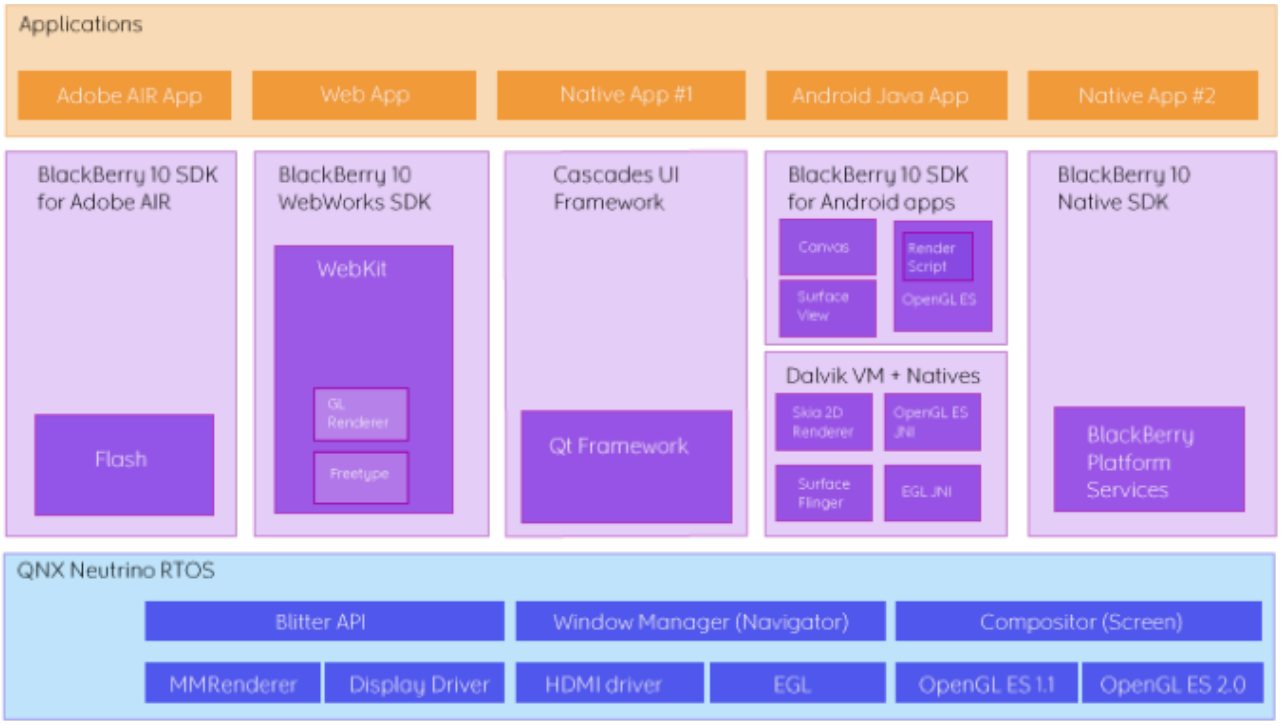


*Рисунок 1.**Ошибка! Текст указанного стиля в документе отсутствует.**.1*

QNT Neutrino RTOS – рівень операційної системи реального часу, що реалізує концепцію мікроядер.

Applications – це рівень додатків.

SDK – рівень інструментів для розробки:

- Adobe AIR (для розробки мобільних та настільних додатків на основі Adobe Flash, ActionScript);
- WebWords (для розробки веб-додатків);
- Native (для розробки «рідних» програм для доступу до різних служб; наприклад, навігація, час та дата,);
- Cascades UI Frameworks (для розробки «рідних» GUI-додатків);
- Android Java (для підтримки програм під Google Android та роботи з Java-мідлетами).

Платформа iOS та її набір розробника iOS SDK включає набір фреймворків для взаємодії з апаратними компонентами пристроїв (наприклад: GPS-датчик; бібліотеки для роботи зі звуком, анімацією.

Архітектура iOS платформи:

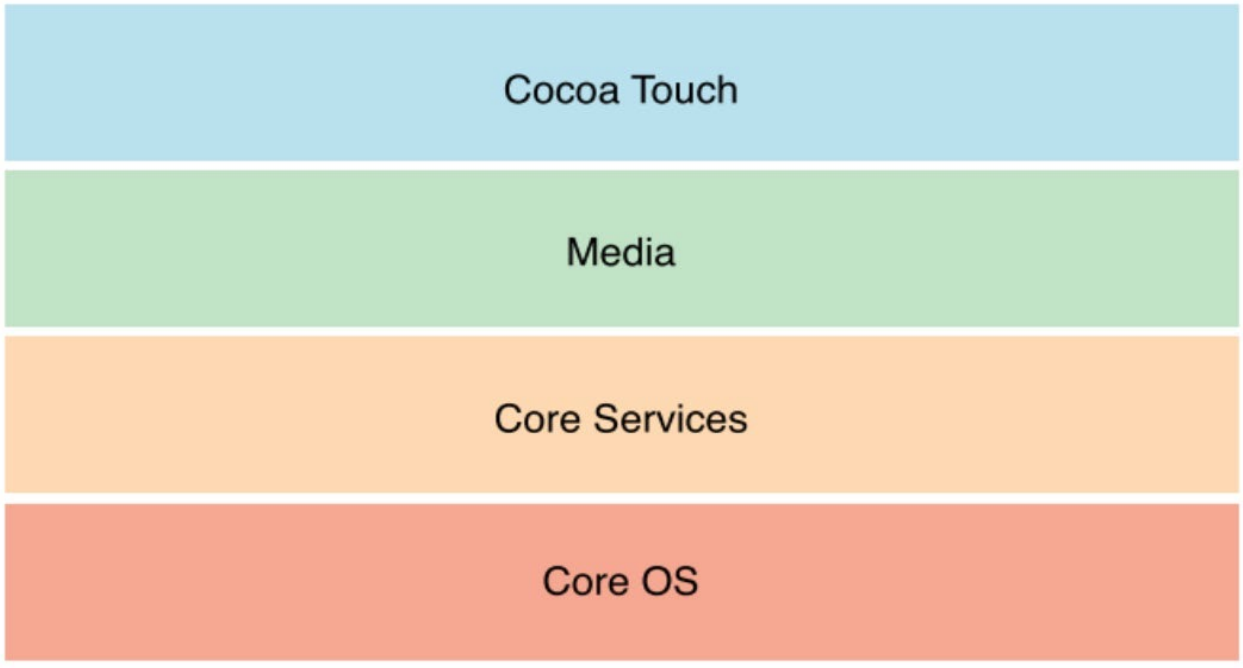


*Рисунок 1.**Ошибка! Текст указанного стиля в документе отсутствует.**.2*

Шар Cocoa Touch включає фреймворки для побудови iOS додатків (для формування зовнішнього вигляду програм, інфраструктури додатків при використанні багатозадачності, сенсорного введення, пуш-повідомлень і доступу на високому рівні до інших системних сервісів).

Шар Media містить технології для роботи з аудіо, відео та графікою.

Шар Core Service містить фундаментальні системні сервіси для програм (Core Foundation та Foundation фреймворки, які визначають базові типи, що використовуються при розробці; технології для роботи з локаційними сервісами, iCloud, соціальними мережами та медіа).

Шар Core OS містить низькорівневі інструменти, на яких будується більшість технологій (забезпечення безпеки або взаємодії із зовнішніми апаратними доповненнями).

1.3 Середовища розробки мобільних додатків

У перші роки розвитку мобільних додатків єдиним способом гарантувати оптимальну роботу додатка на будь-якому пристрої була його власна розробка. Це означало, що новий код потрібно було написати спеціально для конкретного процесора кожного пристрою. Сьогодні більшість розроблених мобільних додатків не залежать від пристрою.

У минулому, якщо програмі потрібно було бути кросплатформеною та працювати на кількох операційних системах (ОС), було небагато, якщо взагалі було, коду, який можна було повторно використати з початкового проекту розробки. По суті, для кожного пристрою потрібен був власний проект розробки мобільного додатку з власною кодовою базою. Сучасні кросплатформні інструменти використовують загальні мови, такі як C# і JavaScript, для спільного використання коду між проектами; що більш важливо, вони добре інтегруються з інструментами керування життєвим циклом програми, такими як Jenkins. Це дозволяє розробникам використовувати єдину кодову базу для Apple iOS, Google Android і прогресивних веб-програм (PWA). PWA створено для використання переваг власних функцій мобільного пристрою, не вимагаючи від кінцевого

користувача відвідувати магазин додатків, робити покупки та завантажувати програмне забезпечення локально. Натомість PWA можна знайти за допомогою запиту в пошуковій системі та отримати миттєвий доступ через браузер, таким чином усуваючи потребу продавцям електронної комерції розробляти рідні програми для кількох мобільних ОС. Основними характеристиками засобів розробки є:

1) доступність (безкоштовна або комерційна);
2) типи пристроїв, під які можлива розробка;
3) підтримувані платформи для розробки;
4) мови програмування;
5) наявність інструментів зборки та відкладки.

Середовища розробки під Android

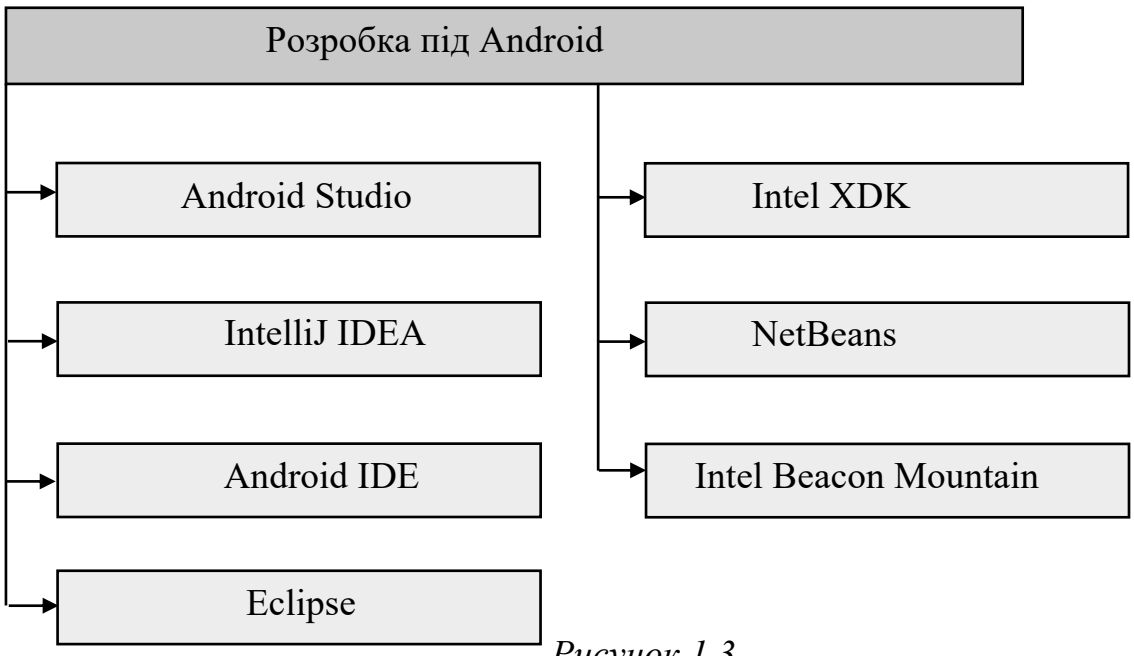


Рисунок 1.3

Android Studio – це інтегроване середовище розробки додатків для Андроїд від Google на основі IntelliJ IDEA

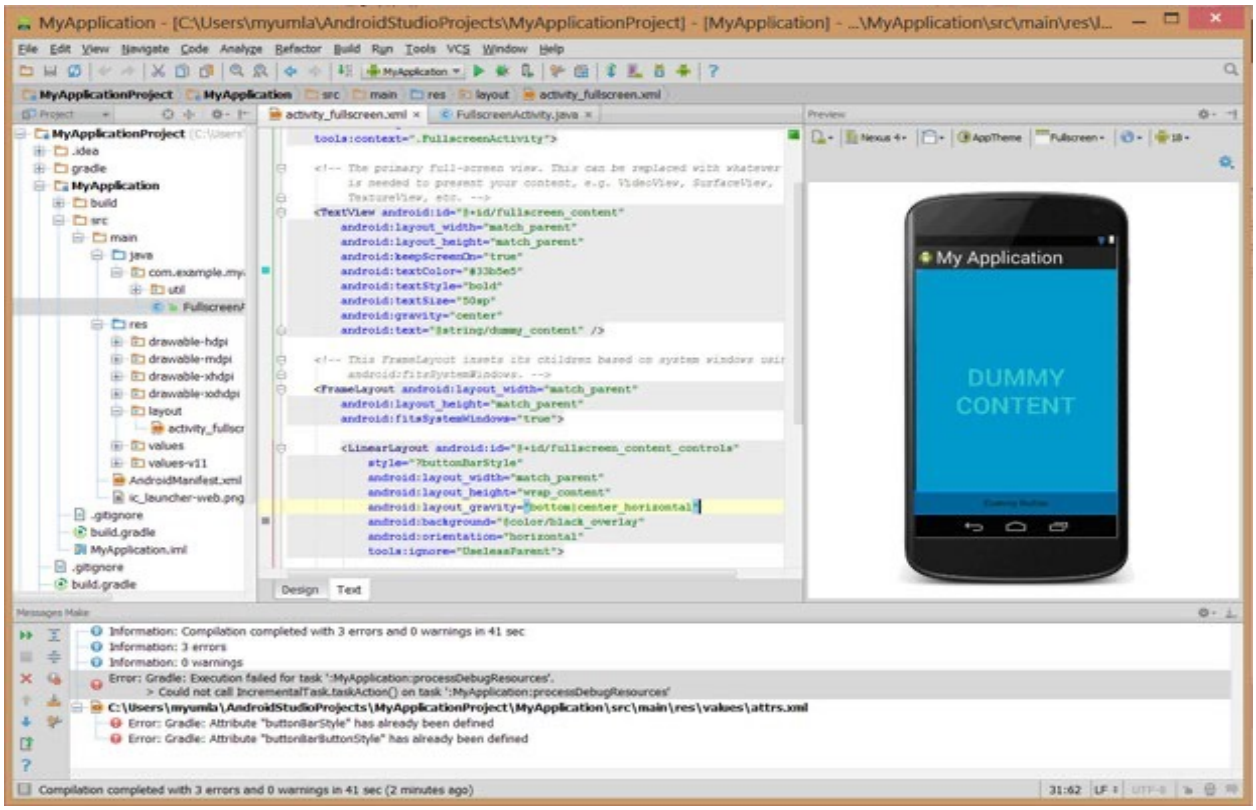


Рисунок 1.4

Середовище надає основні можливості:

- засоби для розробки додатків не тільки для смартфонів і планшетів, але і для пристроїв на базі Android Wear, телевізорів (Android TV), окулярів Google Glass та автомобільних інформаційно-розважальних систем (Android Auto);
- інструмент для автоматичного імпорту існуючого проекту Eclipse (ADT Plugin) у проект Android Studio;
- засоби для спрощення тестування програм на сумісність із різними версіями платформи;
- інструменти для проектування додатків, що працюють на пристроях з різними дозволами екрану (планшети, смартфони, ноутбуки, годинники, окуляри і т.п.).

Також є додаткові можливості:

- складання програми, заснована на Gradle;
- специфічний рефакторинг та швидке виправлення дефектів;

- інструменти для пошуку проблем з продуктивністю, зручністю використання, із сумісністю версій та інших;
- утиліти для скорочення, оптимізації коду, а також цифрового підпису додатків;
- засоби для створення загальних Android конструкцій та компонентів;
- редактор, що працює на багатьох розмірах екранів та дозволів, вікно попереднього перегляду, що показує запущену програму відразу на декількох пристроях і в реальному часі;
- інтерфейс перекладу іншими мовами.

Android IDE – середовище розробки під Android, засноване на Eclipse.

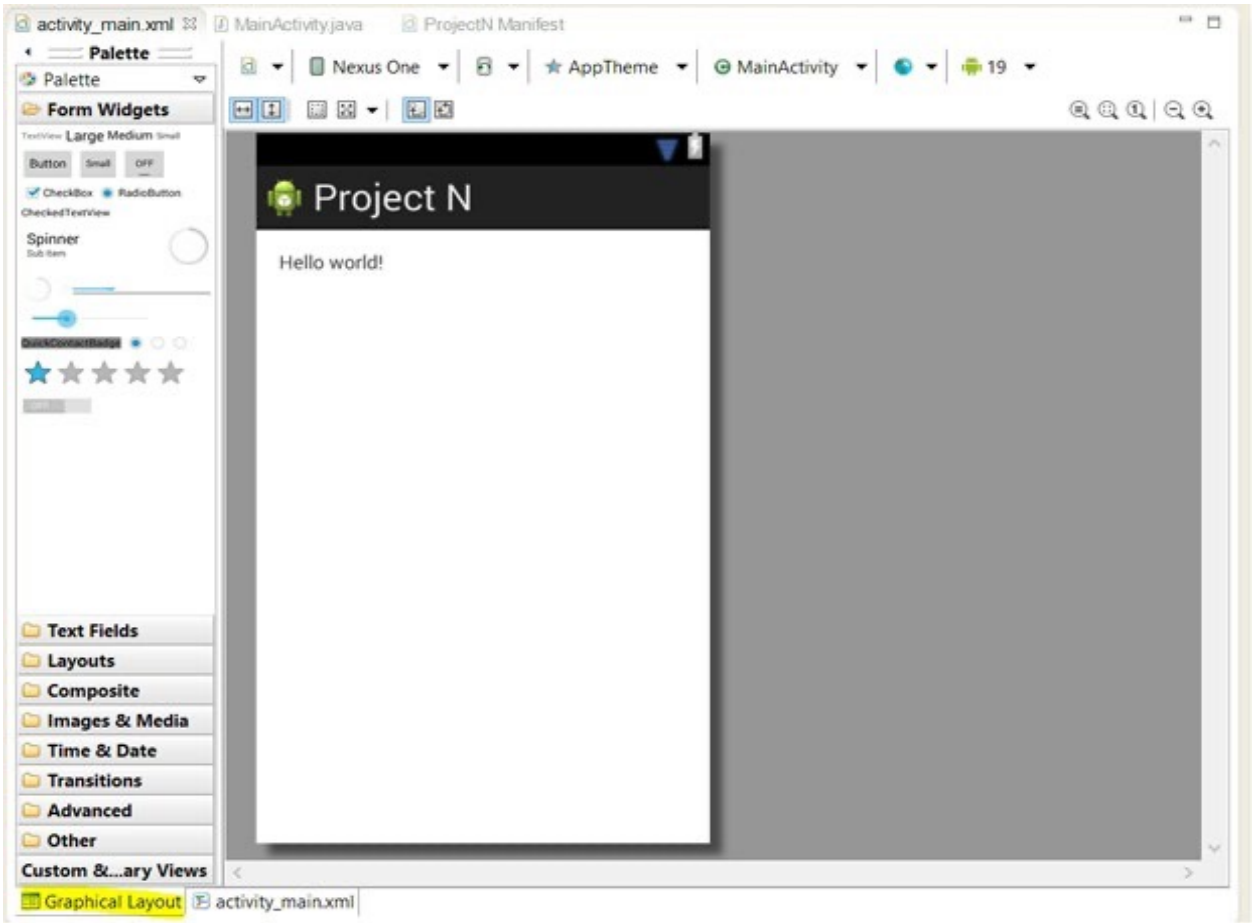


Рисунок 1.5

Містить в собі наступний функціонал:

- засоби розробки (автодоповнення коду, перевірка помилок у реальному часі, рефакторинг тощо);
- менеджер SDK для керування версіями SDK;

- емулятор для розробки та тестування мобільних додатків без залучення реальних пристроїв;
- менеджери віртуальних пристроїв (інструменти для створення та управління віртуальними пристроями в Android (AVD) у вигляді окремих екземплярів емулятора);
- інструмент, що надає графічний інтерфейс до кількох інструментів, призначених для аналізу та налагодження додатків Android;
- засіб розробки на C/C++ та Android NDK.

Середовище розробки Intel XDK дозволяє легко розробляти крос-платформні мобільні програми.

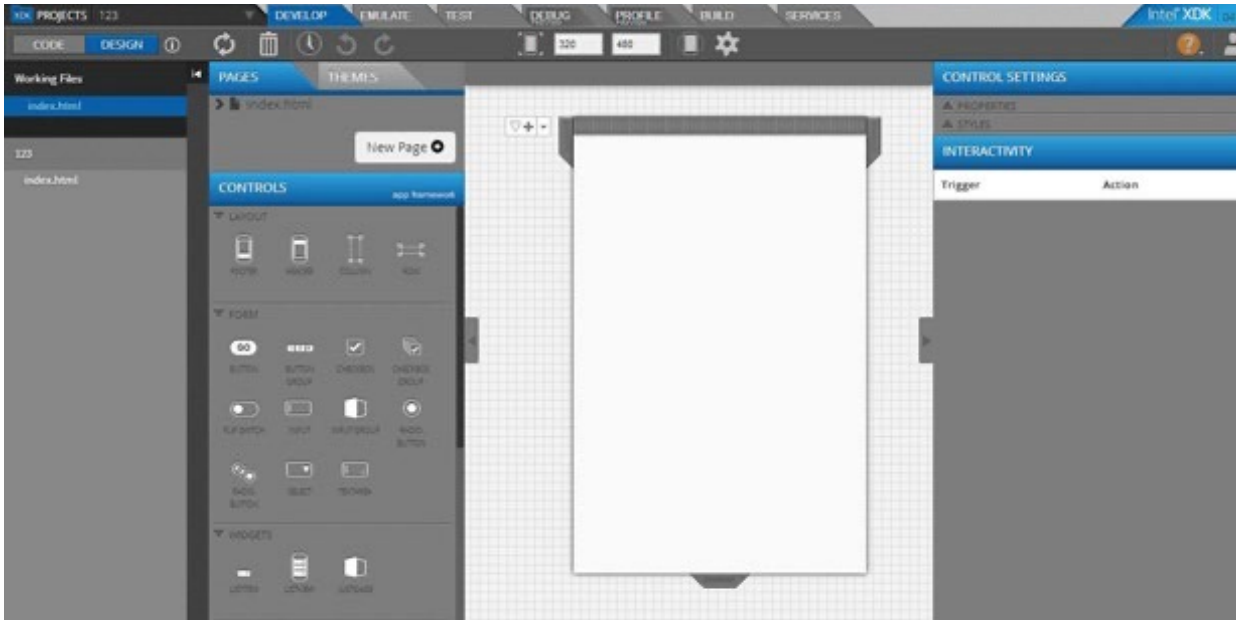


Рисунок 1.6

Intel XDK надає такі можливості:

- засоби розробки (редактор, емулятор пристроїв та відладчик);
- бібліотеку інтерфейсів Javascript, оптимізовану для мобільних додатків;
- додаток для тестування на пристроях;
- прикладний програмний інтерфейс для ігрових програм з
- прискореною візуалізацією;
- систему Intel на базі хмари для підготовки версій програм для більшості Інтернет-магазинів;
- засіб перенесення додатків iOS у середу HTML5;

- набір функцій для взаємодії з операційною системою пристрою (підключення нативних плагінів, роботу з контактами телефону, камерою, геолокацією, вбудованими відео- та аудіоплеєрами і т.д.).

Intel Beacon Mountain – набір інструментів для проектування, розробки, налагодження та оптимізації додатків для Android (побудований на основі Android IDE). Підтримує розробку для цільових платформ на основі процесорів Intel Atom та ARM, що містить:

- процесор віртуалізації для прискорення роботи емулятора серед розробки (Intel* Hardware Accelerated Execution Manager (Intel* HAXM);
- аналізатор продуктивності, що дозволяє оптимізувати завантаженість системи при використанні процедур OpenGL (Intel Graphics Performance Analyzers (Intel GPA) System Analyzer);
- бібліотека оптимізованої обробки даних та зображень, що є частиною повної версії Intel IPP (Intel Integrated Performance Primitives (Intel IPP) Preview);
- бібліотека шаблонів C++ для створення додатків, що масштабуються, і збільшення продуктивності за рахунок розпаралелювання (Intel* Threading Building Blocks (Intel* TBB));
- утиліта для встановлення оновлень, підтвердження статусу підписки на програмне забезпечення, активації (Intel* Software Manager).

Середовища розробки під iOS:

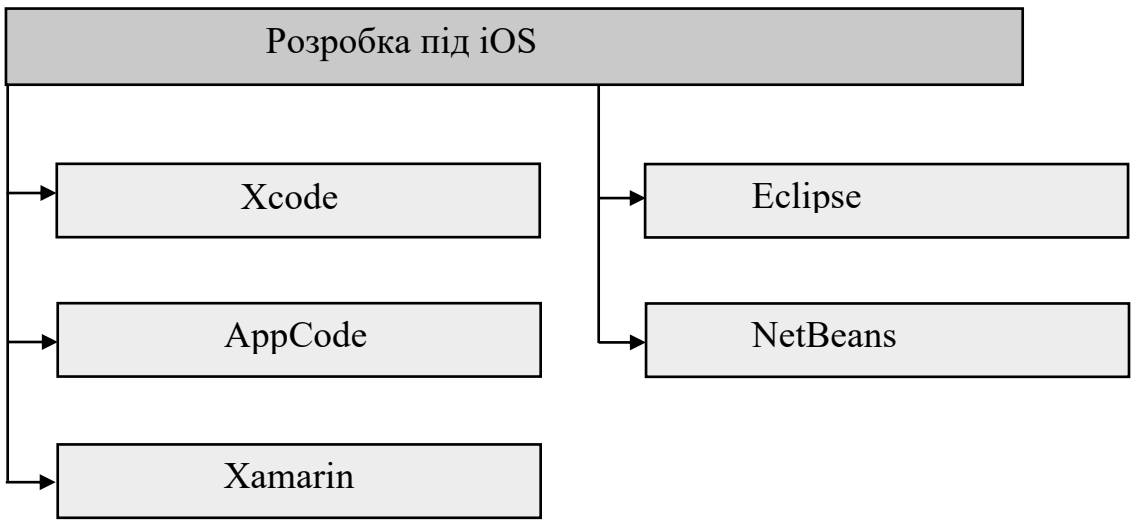


Рисунок 1.7

Xcode – це інтегроване середовище розробки від Apple (IDE).

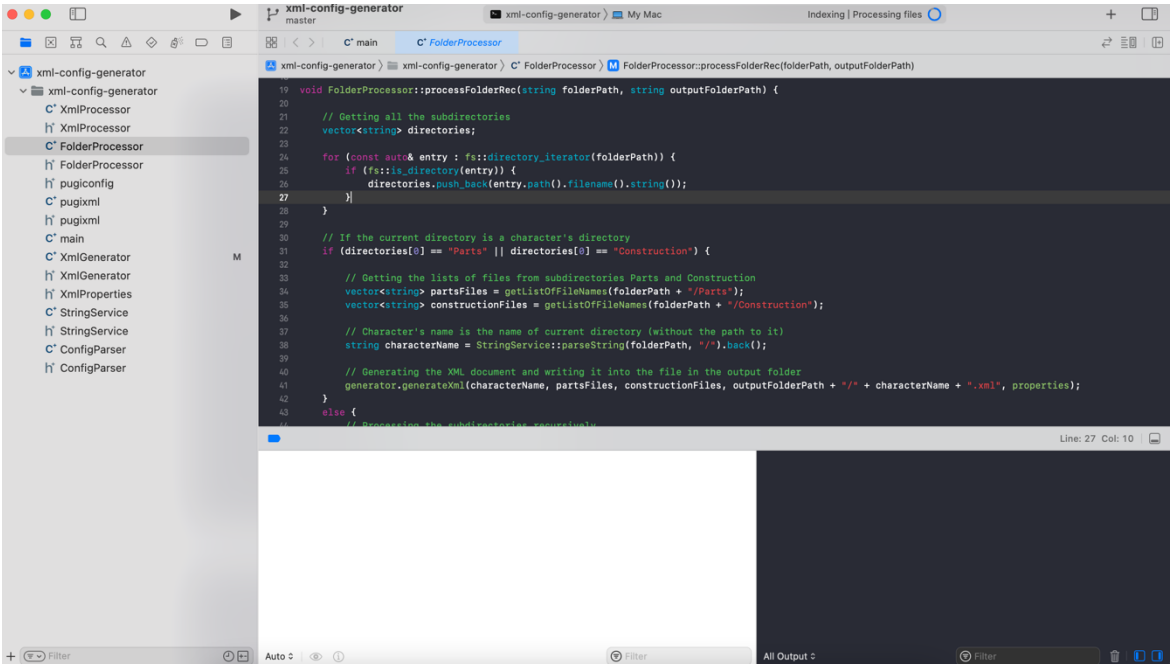


Рисунок 1.8

Включає:

- Редактор вихідного коду (підтримує мови C, C++, Objective-C, Objective-C++, Java, AppleScript, Python, Ruby, Pirl);
- графічний редактор інтерфейсу користувача;
- інтегрований дебагер;
- симулятор різних пристроїв;
- засіб розробки веб-додатків;
- довідкова документація.

AppCode – альтернативне середовище розробки на Objective-C від компанії JetBrains.

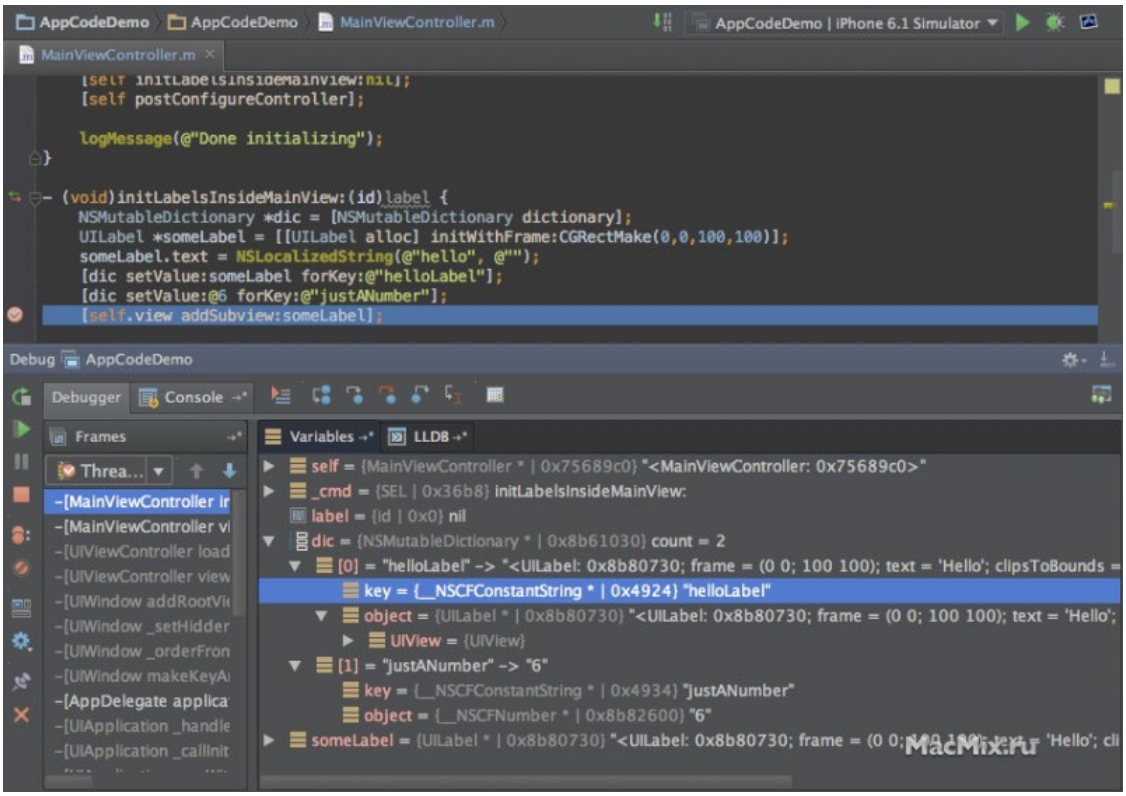


Рисунок 1.9

Відмінності:

- зручна навігація за кодом;
- поліпшена функція автодоповнення;
- автоматизований рефракторинг;
- миттєвий аналізатор коду;
- сумісність проектів з XCode та Interface Builder;
- запуск програм в емуляторі або безпосередньо на пристрої;
- покращений дебагер;
- підтримка систем контролю версій.

Xamarin - це фреймворк для кроссплатформенної розробки мобільних додатків (iOS, Android, Windows Phone) з використанням мови C #.

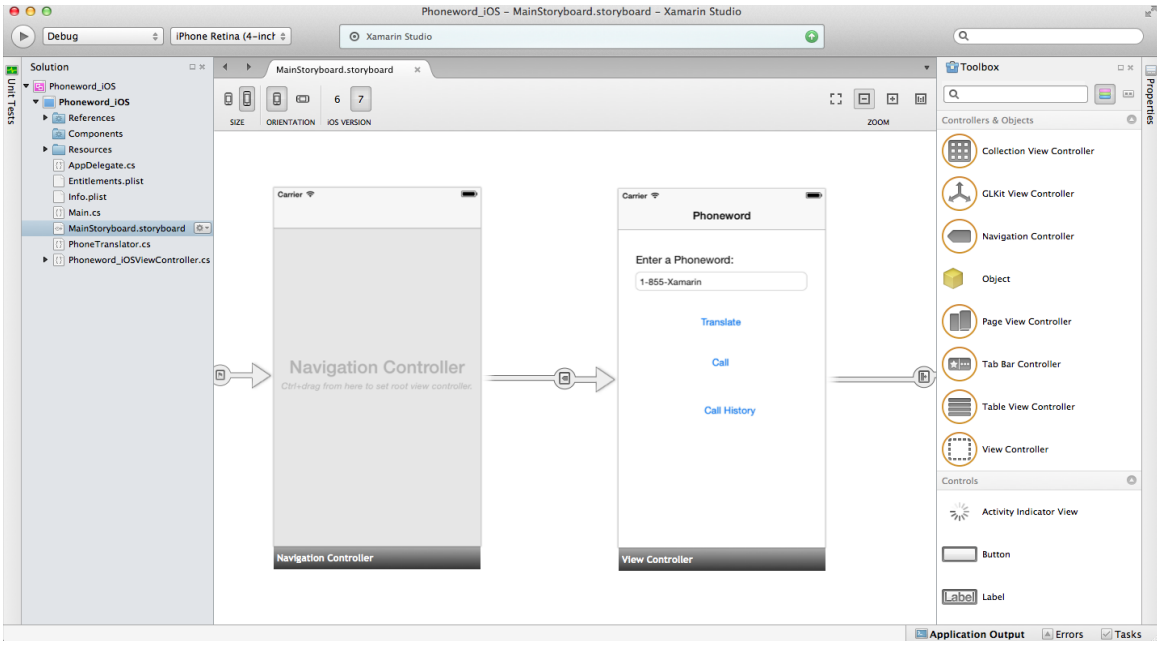


Рисунок 1.10

Фреймворк складається з:

- бібліотеки класів для C#, що надає розробнику доступ до iOS SDK (Xamarin.IOS);

- бібліотеки класів для C#, що надає розробнику доступ до Android SDK (Xamarin. Android);

- компілятори для iOS та Android;

- інтегроване середовище IDE Xamarin Studio (засоби підсвічування, автодоповнення, навігації, аналізу коду, рефакторингу, контролю версій тощо);

- плагін для Visual Studio.

Середовища розробки під Windows Phone:

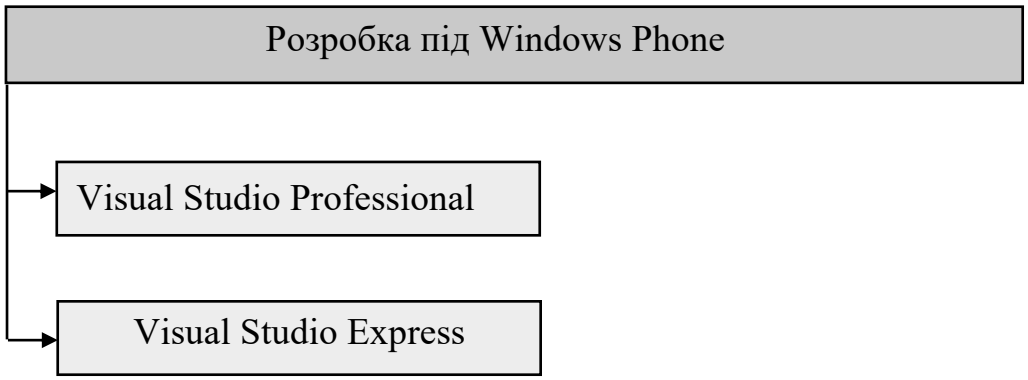


Рисунок 1.11

Visual Studio – це велике середовище для розробки програм під Windows і Windows Phone.

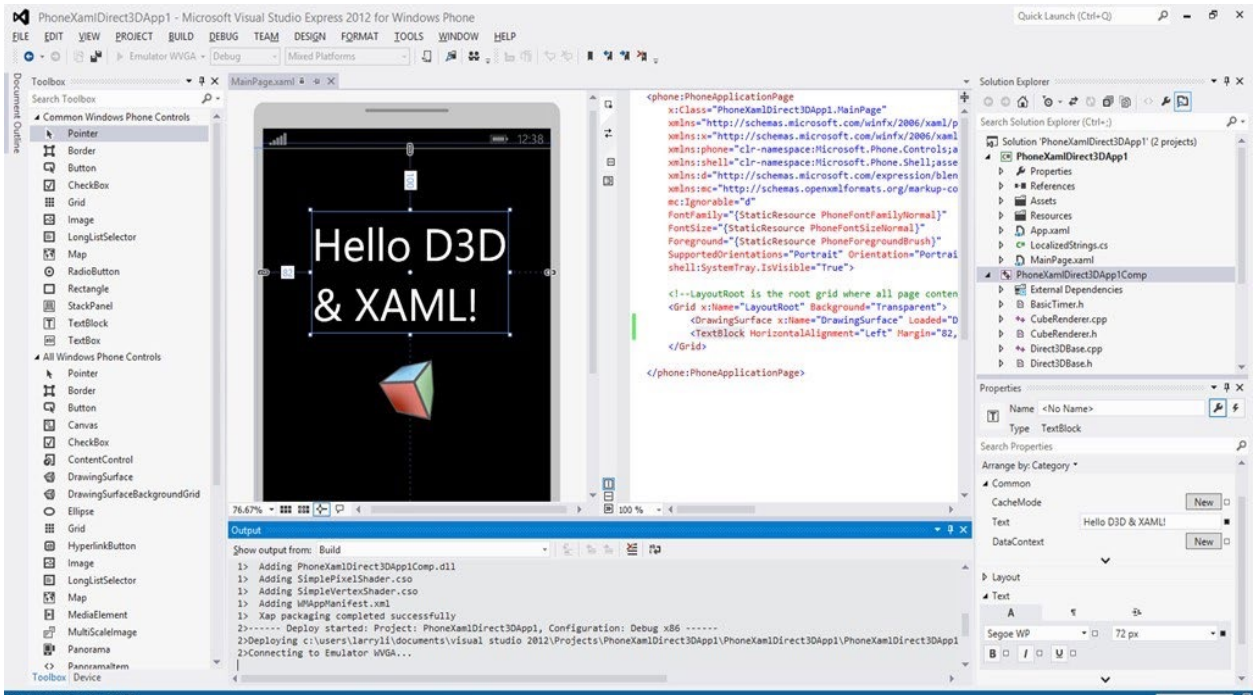


Рисунок 1.12

Має два варіанти комплектації, що містять Windows Phone SDK:

- Visual Studio Professional;
- Visual Studio Express.
- Windows Phone SDK містить:
- Windows Phone SDK;
- Windows Phone Emulator;
- Windows Phone SDK Assemblies;
- Silverlight SDK and DRT (ПЗ для браузера для роботи з мультимедійними даними);
- XNA Game Studio (ПЗ для розробки ігор);
- Expression Blend (інтерактивний візуальний дизайнер для XAML);
- WCF Data Services Client (ПЗ для підтримки обміну даними через Інтернет);
- Microsoft Advertising SDK (для підтримки реклами).

Представлено декількома шаблонами додатків. Windows Phone Application - простий діалоговий додаток, що має один основний екран, через який відбувається основна взаємодія з користувачем

Рисунок 1.13

Windows Phone Pivot Application - програма із закладками, де заголовок кожної закладки визначає вміст. Наприклад, календар, поштовий клієнт та налаштування телефону.

Рисунок 1.14

Windows Phone Panorama Application – програма-панорама, де зони взаємодії з користувачем також розподілені на панелі, але доступні вони через горизонтальне прокручування (фонове зображення встановлено відразу всю панораму, вона має загальний заголовок; контент сусідньої панелі праворуч видно при відображенні поточної).

Рисунок 1.15

Існують середовища, що підтримують декілька платформ. *Жовтим* кольором виділено повністю платні середовища; *помаранчевим* кольором виділені продукти, які мають безкоштовну та платну комплектацію.

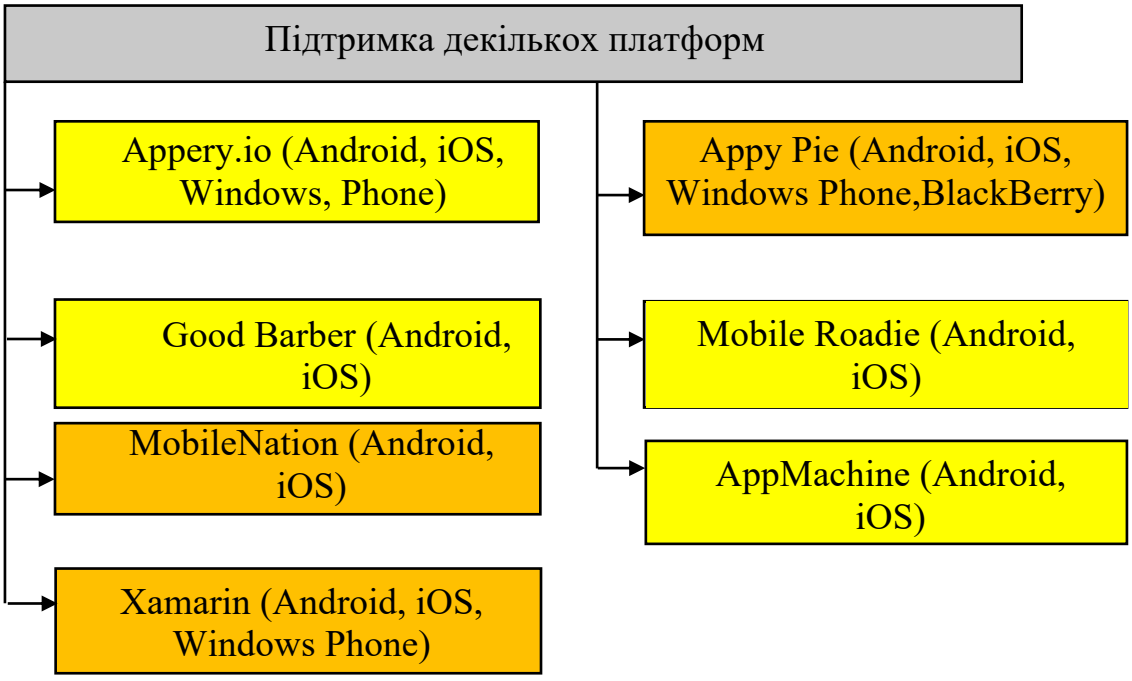


Рисунок 1.16

1.4 Аналіз та перспективи використання Android систем в інформаційних системах

Набір програмного забезпечення, необхідного для розробки кінцевого продукту, буде залежати від безлічі факторів, що впливають на мінімальний набір програмного забезпечення, так і на додатки для нього, що забезпечують саму можливість чи зручність використання. Серед цих факторів:

- платформа, під яку розробляється програма;
- мови програмування, які для цього використовуються;
- бібліотеки та фреймворки, необхідні для вирішення поставлених завдань;
- необхідність проектування структури додатка, його складність та глибина деталізації;
- необхідність створення елементів графічного інтерфейсу.

Цільова платформа, що розробляється, має операційну. систему Android і не має залежностей від будь-яких існуючих ресурсів, будь то програми для настільних платформ або веб-сайтів. Таким чином, програма, що розробляється, може бути класифікована як нативний додаток для мобільних пристроїв під керуванням операційної системи Android. Інші мобільні операційні системи не використовуються замовником у роботі, що визначає безліч інструментів, що використовуються розробки . Розробка під обрану мобільну платформу можлива на безлічі мов програмування, деякі з яких є основними, чи офіційними, мовами, підтримка інших реалізована сторонніми розробниками. До першої групи належать Java, C++, Котлін. До другої можна віднести мови Python (проект Kivy), C# (Xamarin), Basic (B4X).

Використання інших проектів, таких як Xamarin або Kivy, можуть мати проблеми з розміром файлів програми та складністю оптимізації під цільову платформу. Водночас корпорація Google є розробником Android, оголосила мову Kotlin офіційно підтримуваним у їхній операційній системі у 2017 році. Kotlin має можливість працювати на основі Java Virtual Machine, що дозволяє використовувати продукти екосистеми Java, але відрізняється лаконічністю синтаксису, на відміну Java. Kotlin, при цьому, захищений від NullPointerException

і пов'язані з цим проблеми в тестуванні. Для етапу розробки та проектування також цінні класи даних та перерахувань, спрямованими на написання більш компактного коду та структурування відносин між даними за допомогою елементів перерахування як маркерів типу замість перевантаженої системи успадкування. Таким чином, Kotlin є кращою мовою для розробки під Android ніж Java. C++ хоч і є частиною екосистеми Android, але використовується для специфічних завдань, до яких певна у межах даної роботи не належить.

Фреймворки є інструментами для спрощення роботи з колом часто зустрічаються завдань, або ж рішеннями специфічних питань, компенсуючими недоліки мови чи платформи. Вибір фреймворків для розробки програмного продукту, а також вибір менш комплексних рішень бібліотек, обумовлений фактором тимчасової вигоди, що відбувається також від рівня володіння відповідальними розробниками інструментом. У разі розробки даного мобільного додатка, виходячи з навичок розробника та передбачуваної моделі розробки, передбачає найшвидший реліз першої працездатної версії (Робочого прототипу), немає необхідності у використанні будь-яких фреймворків. Для мобільної операційної системи Android, залежно від класифікації, існує три або чотири варіанти роботи з даними із додатку: зовнішнє та внутрішнє сховища (як окремі файли), Shared Preferences (збережені налаштування програми), система управління базами даних SQLite (файл бази даних). Для основного функціоналу програми (роботи із замовленнями) підходить останній варіант, коли дані зберігаються в базі даних під керуванням SQLite, оскільки позначені вище можливості мови Kotlin (а саме Data Class) добре переносяться на базу даних при її структурному проектуванні. Для зберігання цін та коефіцієнтів для розрахунку кінцевої вартості замовлень та для логічного відокремлення даних параметрів від самих параметрів замовлень передбачається використовувати Shared Preferences як спосіб їх зберігання. При реалізації експериментальних функцій, пов'язаних із зберіганням текстових нотаток, варіант із зберіганням у вигляді окремих файлів є адекватним, бо дозволяє, за потреби, модернізувати внутрішню структуру текстових файлів для

створення шаблонів для різних типів нотаток та інших маніпуляцій із текстом, що зберігається в них.

Таким чином, всі зазначені на початку пункту 1.2.1 фактори що впливають на вибір програмного забезпечення та технологій, які були розглянуті. Як інструменти були обрані Android Studio в ролі IDE, а для роботи з графічними завданнями – Kotlin. Після вибору інструментів було дано опис архітектури розробки на основі поставлених завдань, вибраних інструментів та особливостей розробки.

2 СТРУКТУРА ANDROID ДОДАТКІВ

2.1 Структура додатків під платформу Android

Додаток на Android це такий собі Спеціальний файл-архів із розширенням *.apk,*який являє собою набір компонентів. Додаток немає точки входу як додаток на java, тобто. немає методу main(). Програма представлене через контекст, тобто кожна програма виконується в окремому процесі під окремим користувачем у розрахованому на багато користувачів середовищі з мінімальними привілеями.

Особливості таких додатків:

1) одна програма може використовувати компоненти інших додатків (якщо ці програми дозволяють їх використовувати);
2) додаток не містить код інших додатків та посилання на них;
3) система повинна запустити процес додатку, у якому знаходиться необхідний елемент, і ініціалізувати потрібні йому об'єкти.

Архітектура Android-додатки є фреймворк-орієнтованою (framework-based), тобто, зводиться до розширення деяких класів чи реалізації інтерфейсів, наданих фреймворком. Така програма не може бути запущена поза фреймворком або без нього.

Унікальна система управління пам'яттю:

- збирач сміття може знищити об'єкти, з якими поточний час немає взаємодії, якщо ОС вирішить, що мало;
- ОС може знищити процес, який в даний момент не показує користувачеві жодного графічного інтерфейсу.

Обробка приховування інтерфейсів користувача та існування кнопки «назад» на Android пристроях призводить до необхідності наявності стека користувальницьких інтерфейсів, в якому поточний видимий інтерфейс поміщається на вершину, а всі інші зсуваються вниз.

Обробка приховування інтерфейсів користувача та існування кнопки «назад» на Android пристроях призводить до необхідності наявності стека користувальницьких інтерфейсів, в якому поточний видимий інтерфейс поміщається на вершину, а всі інші зсуваються вниз.

Обробка взаємодії між інтерфейсом користувача та його логікою буде слідувати архітектурному шаблону «Model-View-ViewModel» (MVVM, Модель-Представлення-Модель представлення).

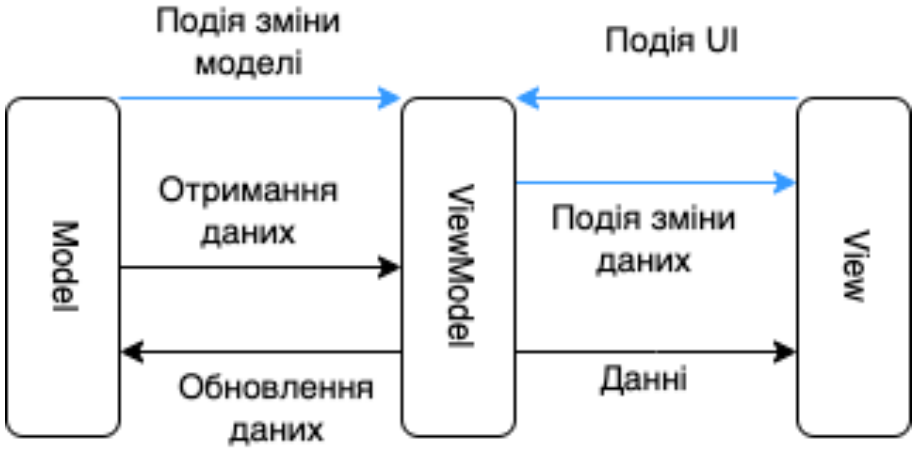


Рисунок 2.1

Інтерфейс користувача реалізується гіпертекстової розміткою (XML). Логіка інтерфейсу користувача реалізується розробником як компонент ViewModel.

Функціональні зв'язки між інтерфейсом користувача і ViewModel реалізуються через біндинги (bindings), які є правилами типу «якщо кнопка А була натиснута, повинен бути викликаний метод onButtonAClick() з ViewModel».

2.2 Компоненти додатків Android

Компоненти програми є основними будівельними блоками програми Android. Кожен компонент є точкою входу, через яку система або користувач може увійти у програму. Деякі компоненти залежать від інших.

Є чотири різні типи компонентів програми:

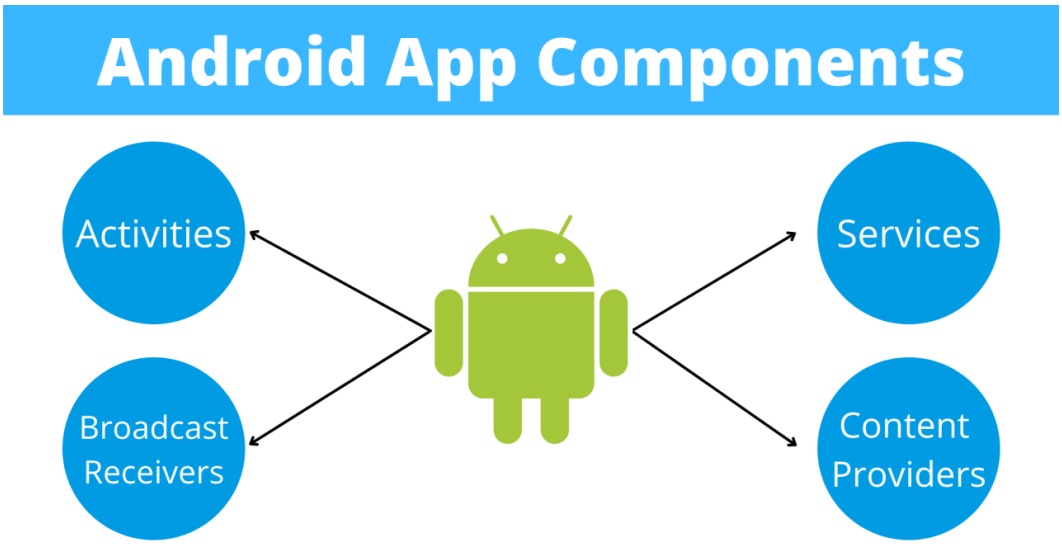


Рисунок 2.2

Кожен тип служить певній меті та має окремий життєвий цикл, який визначає, як компонент створюється та знищується. Розберемо кожний більш детально.

Діяльність (Activities). *Діяльність* є точкою входу для взаємодії з користувачем. Він являє собою єдиний екран з інтерфейсом користувача. Наприклад, програма електронної пошти може мати одну дію, яка показує список нових електронних листів, іншу дію для створення електронного листа та іншу дію для читання електронних листів. Незважаючи на те, що дії працюють разом, щоб створити згуртовану взаємодію з користувачем у програмі електронної пошти, кожна з них не залежить від інших. Таким чином, інша програма може почати будь-яку з цих дій, якщо програма електронної пошти це дозволяє. Наприклад, програма камери може розпочати роботу в програмі електронної пошти, яка створює нове повідомлення, щоб дозволити користувачеві поділитися зображенням. Activity сприяє наступним ключовим взаємодіям між системою та програмою:

Відстеження того, що зараз хвилює користувача (те, що на екрані), щоб переконатися, що система продовжує виконувати процес, який розміщує дію.

Знаючи, що процеси, які використовувалися раніше, містять речі, до яких користувач може повернутися (зупинені дії), і, отже, надають більшого пріоритету збереженню цих процесів.

Допомога додатку впоратися з припиненням процесу, щоб користувач міг повернутися до дій із відновленням попереднього стану.

Забезпечення способу для додатків реалізовувати потоки користувачів між собою, а для системи координувати ці потоки. Найбільш класичним прикладом є спільний доступ.

Services — це точка входу загального призначення для підтримки роботи програми у фоновому режимі з усіх причин. Це компонент, який працює у фоновому режимі для виконання тривалих операцій або виконання роботи для віддалених процесів. Services не надає інтерфейс користувача. Наприклад, Services може відтворювати музику у фоновому режимі, поки користувач перебуває в іншій програмі, або вона може отримувати дані через мережу, не блокуючи взаємодію користувача з діяльністю. Інший компонент, наприклад активність, може запустити Services та дозволити їй працювати або прив’язатися до неї, щоб взаємодіяти з нею.

Існує два типи Services, які повідомляють системі, як керувати програмою: Started services та Bound services.

Started services повідомляють системі про те, щоб вони працювали до завершення роботи. Це може бути синхронізація деяких даних у фоновому режимі або відтворення музики навіть після того, як користувач вийде з програми. Синхронізація даних у фоновому режимі або відтворення музики також представляють два різні типи запущених служб, які змінюють спосіб їх обробки системою:

- відтворення музики — це те, про що користувач безпосередньо знає, тому програма повідомляє системі про це, кажучи, що хоче бути на передньому плані зі сповіщенням, щоб повідомити про це користувачеві; у цьому випадку система знає, що їй слід дуже докладати зусиль, щоб процес цієї служби працював, тому що користувач буде незадоволений, якщо вона зникне;
- звичайна фонова служба — це не те, про що користувач безпосередньо знає, що працює, тому система має більше свободи в управлінні своїм процесом. Це може дозволити його вимкнути (а

потім перезапустити службу через деякий час), якщо йому потрібна оперативна пам’ять для речей, які мають більш безпосереднє значення для користувача.

Bound services - запускаються, якщо якась інша програма (або система) заявила, що хоче використовувати службу. По суті, це служба, яка надає API для іншого процесу. Таким чином, система знає, що між цими процесами існує залежність, тому, якщо процес А прив’язаний до служби в процесі В, вона знає, що їй потрібно підтримувати роботу процесу В (і його служби) для А. Крім того, якщо процес А є чимось про що дбає користувач, тоді він також знає, що розглядати процес В як щось, що також хвилює користувача.

Завдяки своїй гнучкості (на краще чи гірше) служби виявилися дійсно корисним будівельним блоком для всіх типів системних концепцій вищого рівня. Живі фонові малюнки, сповіщення, екранні заставки, методи введення, служби доступності та багато інших основних системних функцій створено як служби, які реалізують програми, і система зв’язується з ними, коли вони мають бути запущені.

Broadcast receivers — це компонент, який дозволяє системі передавати події в програму поза звичайним потоком користувача, дозволяючи програмі відповідати на загальносистемні широкомовні оголошення. Оскільки Broadcast receivers є ще одним чітко визначеним записом у програмі, система може доставляти трансляції навіть до програм, які зараз не запущені. Так, наприклад, програма може запланувати нагадування, щоб опублікувати сповіщення, щоб повідомити користувача про майбутню подію... і, доставляючи це нагадування BroadcastReceiver програми, немає необхідності, щоб програма продовжувала працювати, доки не спрацьовує будильник. Багато трансляцій надходять із системи наприклад, трансляція сповіщає про те, що екран вимкнено, акумулятор розряджений або зроблено фотографію. Програми також можуть ініціювати трансляції, наприклад, щоб повідомити іншим програмам, що деякі дані завантажено на пристрій і доступні для використання. Хоча Broadcast receivers не відображають інтерфейс користувача, вони можуть створити сповіщення в рядку

стану, щоб попередити користувача про подію трансляції. Однак найчастіше Broadcast receivers є лише шлюзом до інших компонентів і призначений для виконання мінімального обсягу роботи.

2.3 Активація компонентів та маніфест файли

Activating components, де три з чотирьох типів компонентів — activities, services, and broadcast receivers— активуються асинхронним повідомленням, яке називається - *Intent*. Intents пов'язують окремі компоненти один з одним під час виконання. Розглядати їх як месенджери, які вимагають дії від інших компонентів, незалежно від того, чи належить компонент програмі або ж ні.

Intent створюється за допомогою об'єкта Intent, який визначає повідомлення для активації певного компонента (explicit intent) або певного типу компонента (implicit intent).

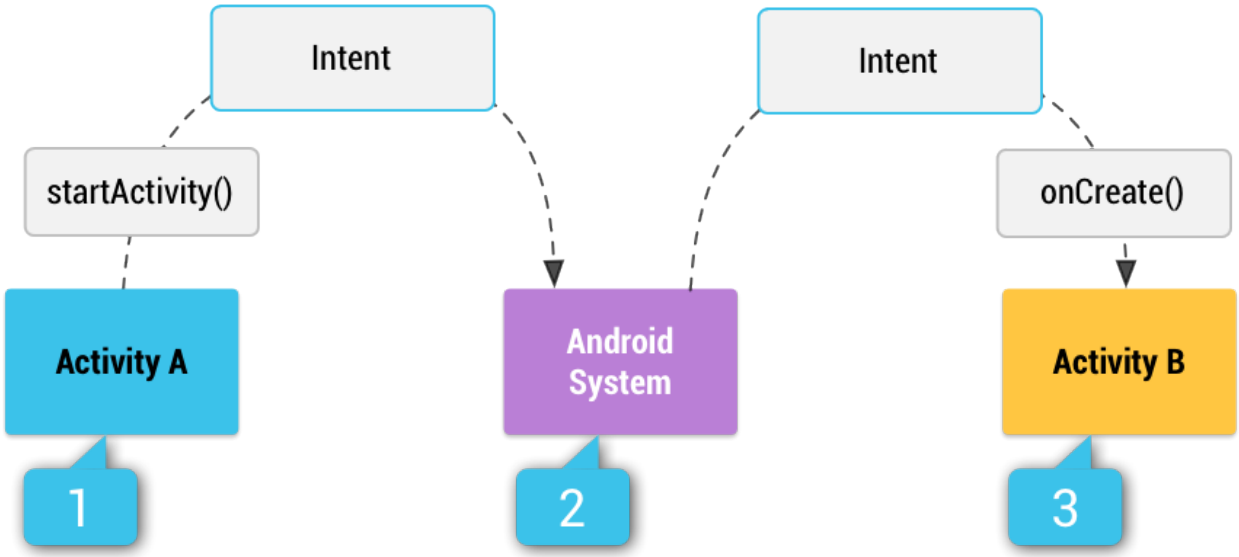


Рисунок 2.3

Для activities and services, Intent визначає дію, яку потрібно виконати (наприклад, переглянути або надіслати щось) і може визначати URI даних, з якими потрібно діяти, серед іншого, що може знадобитися компоненту, який запускається. Наприклад, намір може передати запит на дію, щоб показати зображення або відкрити веб-сторінку. У деяких випадках можливо почати дію, щоб отримати результат, і в цьому випадку дія також повертає результат у Intent.

Наприклад, видати намір дозволити користувачеві вибрати особистий контакт і повернути його вам. Намір повернення включає URI, що вказує на вибраний контакт.

Для broadcast receivers, Intent просто визначає оголошення, яке транслюється. Наприклад, широкомовне повідомлення про низький рівень заряду батареї пристрою включає лише відомий рядок дій, який вказує на низький рівень заряду батареї.

На відміну від activities, services та broadcast receivers, content providers не активуються за допомогою Intents. Навпаки, вони активуються, коли на них націлено запит від ContentResaver. Резолвер вмісту обробляє всі прямі транзакції з постачальником вмісту, тому компонент, який виконує транзакції з постачальником, не потребує цього, а замість цього викликає методи об'єкта ContentResaver. Це залишає рівень абстракції між постачальником вмісту та компонентом, який запитує інформацію (для безпеки).

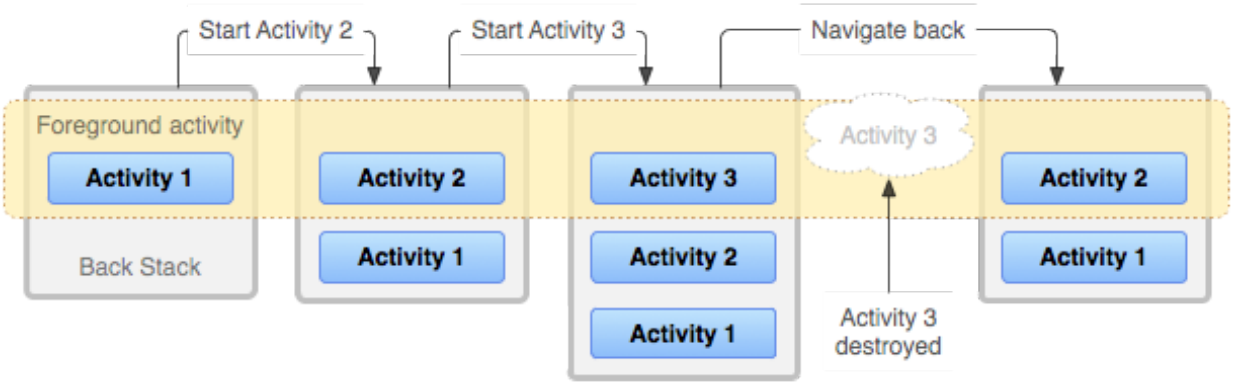


Рисунок 2.4

Існують окремі способи активації кожного типу компонента: розпочати дію або надати їй щось нове, передавши намір у startActivity() або startActivityForResult() (щоб діяльність повернула результат).

В Android 5.0 (рівень API 21) і новіших версіях використовувати клас JobScheduler для планування дій. Для попередніх версій Android запустити службу (або надати нові інструкції поточній службі), передавши Intent до startService(), дає можливість прив'язатися до служби, передавши намір у bindService(), ініціювати

трансляцію, передавши Intent таким методам, як sendBroadcast(), sendOrderedBroadcast() або sendStickyBroadcast(), виконати запит до постачальника вмісту, викликавши query() на ContentResolver.

Щоб отримати додаткові відомості про використання намірів, перегляньте документ «Наміри та фільтри намірів». У наведених нижче документах надається додаткова інформація про активацію окремих компонентів: Activities, Services, BroadcastReceiver, and Content Providers

Файл маніфесту, перш ніж система Android зможе запустити компонент програми, вона має знати, що компонент існує, прочитавши файл маніфесту програми AndroidManifest.xml. Програма має оголосити всі свої компоненти в цьому файлі, який має бути в корені каталогу проекту програми.

Окрім оголошення компонентів програми, маніфест виконує ряд функцій, наприклад:

- визначає будь-які дозволи користувача, які вимагає додаток, наприклад доступ до Інтернету або доступ для читання контактів користувача;
- оголошує мінімальний рівень API, необхідний програмі, на основі того, які API використовує програма;
- оголошує апаратні та програмні функції, які використовуються або потрібні програмі, як-от камера, служби Bluetooth або мультисенсорний екран;
- оголошує бібліотеки API, з якими програму потрібно зв’язати (окрім API фреймворку Android), наприклад бібліотеку Google Maps.

2.4 Використання ресурсів та їх керування в Android додатках

Додаток для Android складається не лише з коду — для нього потрібні ресурси, відокремлені від вихідного коду, як-от зображення, аудіофайли та все, що стосується візуального представлення додатка.

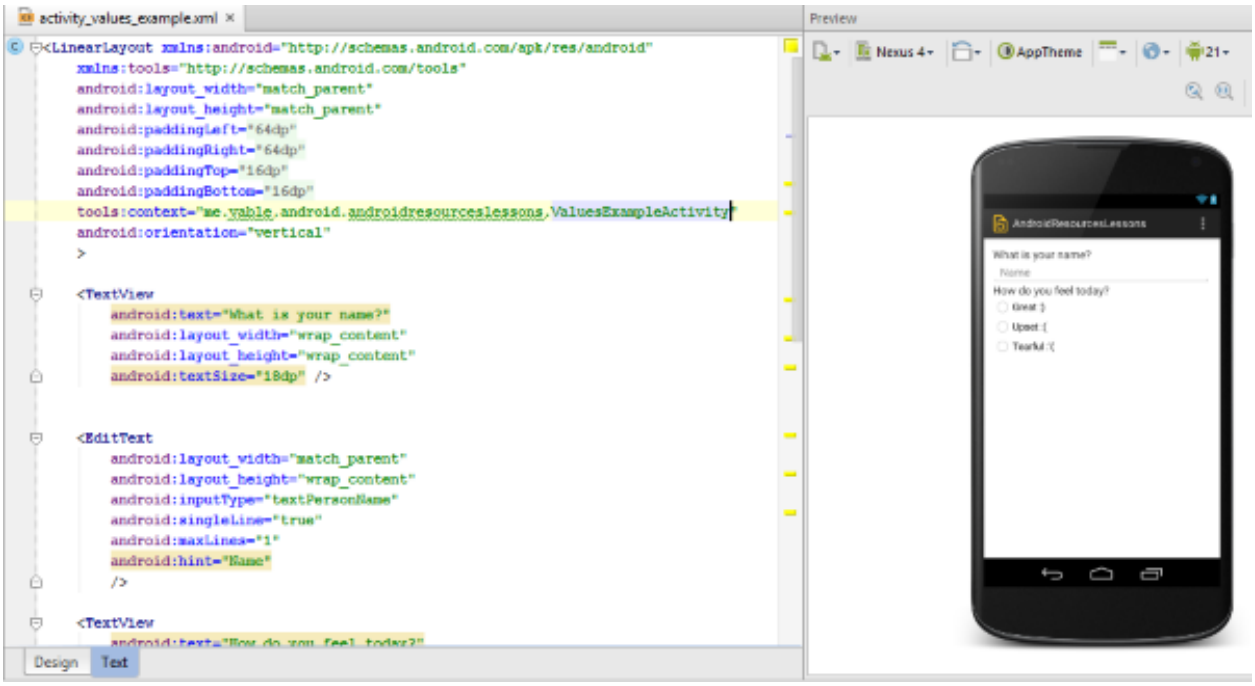


Рисунок 2.5

Наприклад, визначити анімацію, меню, стилі, кольори та компонування інтерфейсів користувача за допомогою файлів XML.

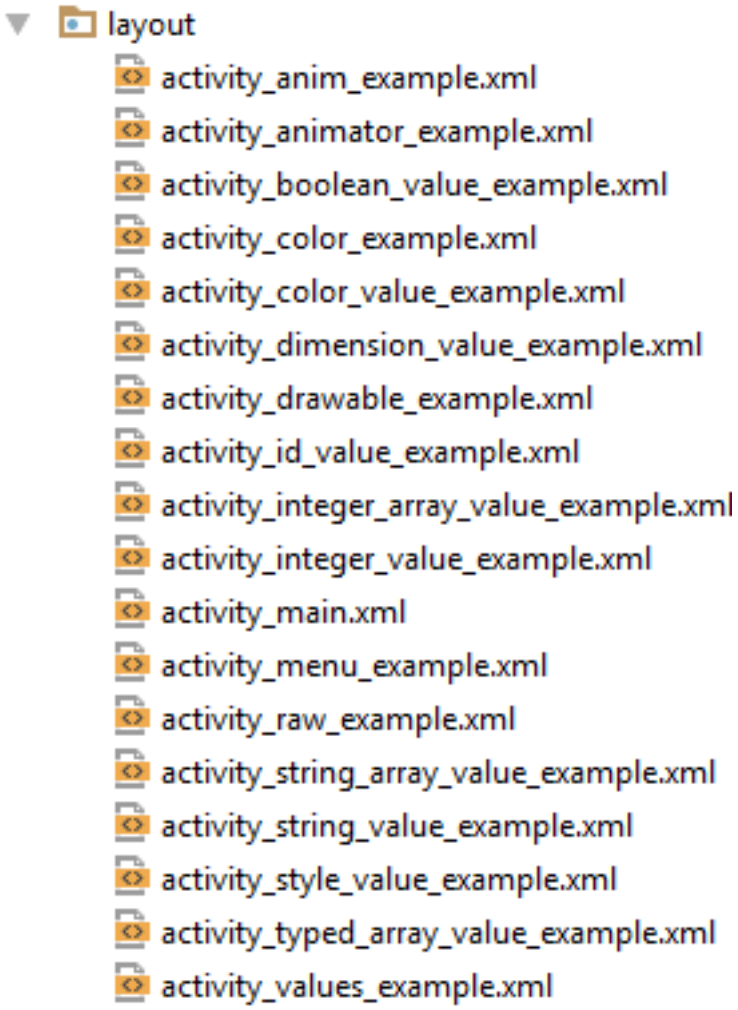


Рисунок 2.6

Використання ресурсів програми дозволяє легко оновлювати різні характеристики програми, не змінюючи код. Надання наборів альтернативних ресурсів дає змогу оптимізувати програму для різних конфігурацій пристроїв, наприклад для різних мов і розмірів екрана.

Рисунок 2.7

Для кожного ресурсу, який включений у проект Android, інструменти створення SDK визначає унікальний цілочисельний ідентифікатор, який можна використовувати для посилання на ресурс із коду програми чи інших ресурсів, визначених у XML. Наприклад, якщо програма містить файл зображення під назвою logo.png (для прикладу збережений у каталозі res/pictures/), інструменти SDK генерують ідентифікатор ресурсу під назвою R.drawable.logo. Цей ідентифікатор зіставляється з цілим числом програми, яке можна використовувати для посилання на зображення та вставити його в інтерфейс користувача.

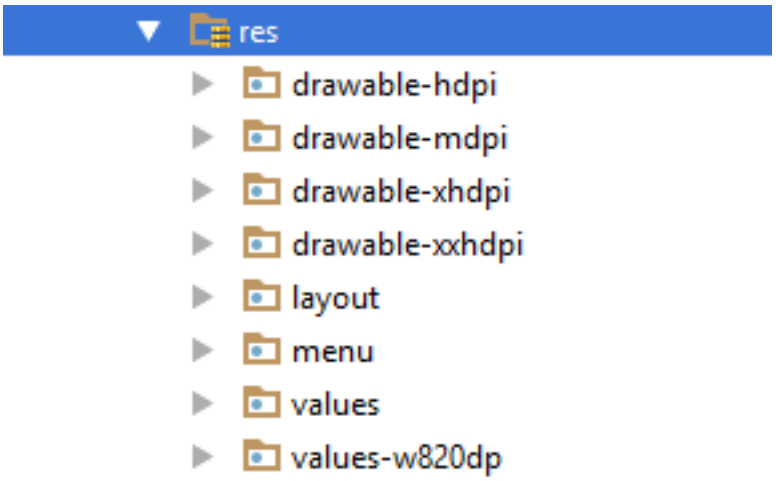


Рисунок 2.8

Одним із найважливіших аспектів надання ресурсів окремо від вихідного коду є можливість надання альтернативних ресурсів для різних конфігурацій пристроїв. Наприклад, визначивши рядки інтерфейсу користувача в XML, є можливість перекладати рядки на інші мови та зберігати ці рядки в окремих файлах. Потім Android застосовує відповідні мовні рядки до інтерфейсу користувача на основі кваліфікатора мови, який додається до назви каталогу ресурсу (наприклад, res/values-fr/ для французьких рядкових значень) і налаштування мови користувача.

Android підтримує багато різних класифікаторів для альтернативних ресурсів. Класифікатор — це короткий рядок, включається в назву своїх каталогів ресурсів, щоб визначити конфігурацію пристрою, для якого ці ресурси повинні використовуватися.

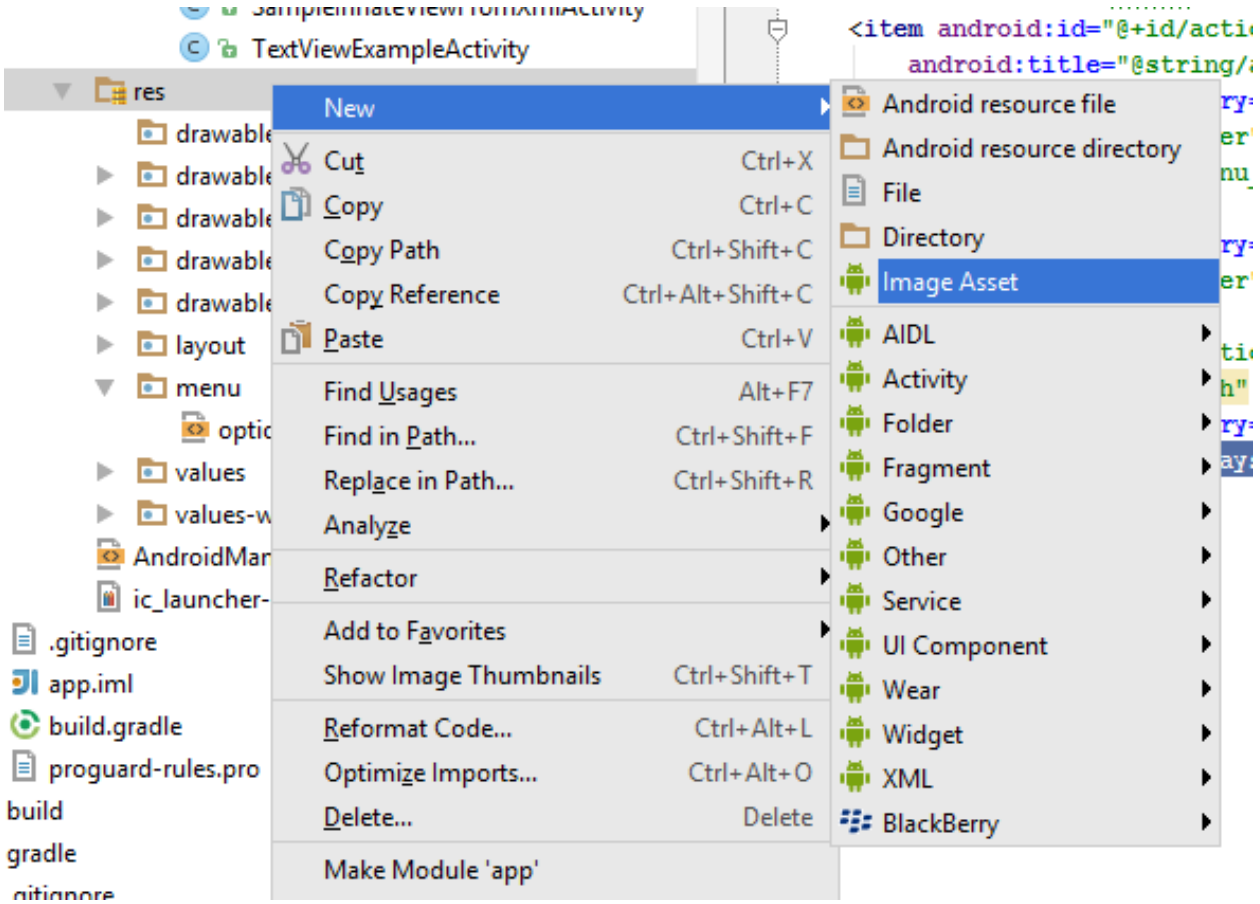


Рисунок 2.9

Наприклад, створити різні макети для своїх дій, залежно від орієнтації екрана та розміру пристрою. Коли екран пристрою має портретну орієнтацію (високий), можливо, щоб макет із кнопками був вертикальним, але коли екран має альбомну орієнтацію (широкий), кнопки можна вирівняти горизонтально. Щоб змінити макет залежно від орієнтації, визначити два різних макета та застосувати відповідний кваліфікатор до імені каталогу кожного макета. Потім система автоматично застосовує відповідний макет залежно від поточної орієнтації пристрою.

3. ПРАКТИЧНА РЕАЛІЗАЦІЯ РОЗШИРЕНИХ МОЖЛИВОСТЕЙ ДЛЯ ДОДАТКУ

3.1. Алгоритм для фільтрування проєктів, новин та подій

Розробка мобільних додатків — це набір процесів і процедур, пов’язаних із написанням програмного забезпечення для невеликих бездротових комп’ютерних пристроїв, таких як смартфони та інші кишенькові пристрої. Як і розробка веб-додатків, розробка мобільних додатків бере свій початок у більш традиційній розробці програмного забезпечення. Проте одна критична відмінність полягає в тому, що мобільні програми часто пишуться спеціально для використання переваг унікальних функцій конкретного мобільного пристрою. На рисунку нижче представлено швидку заміну алгоритмів під час використання.

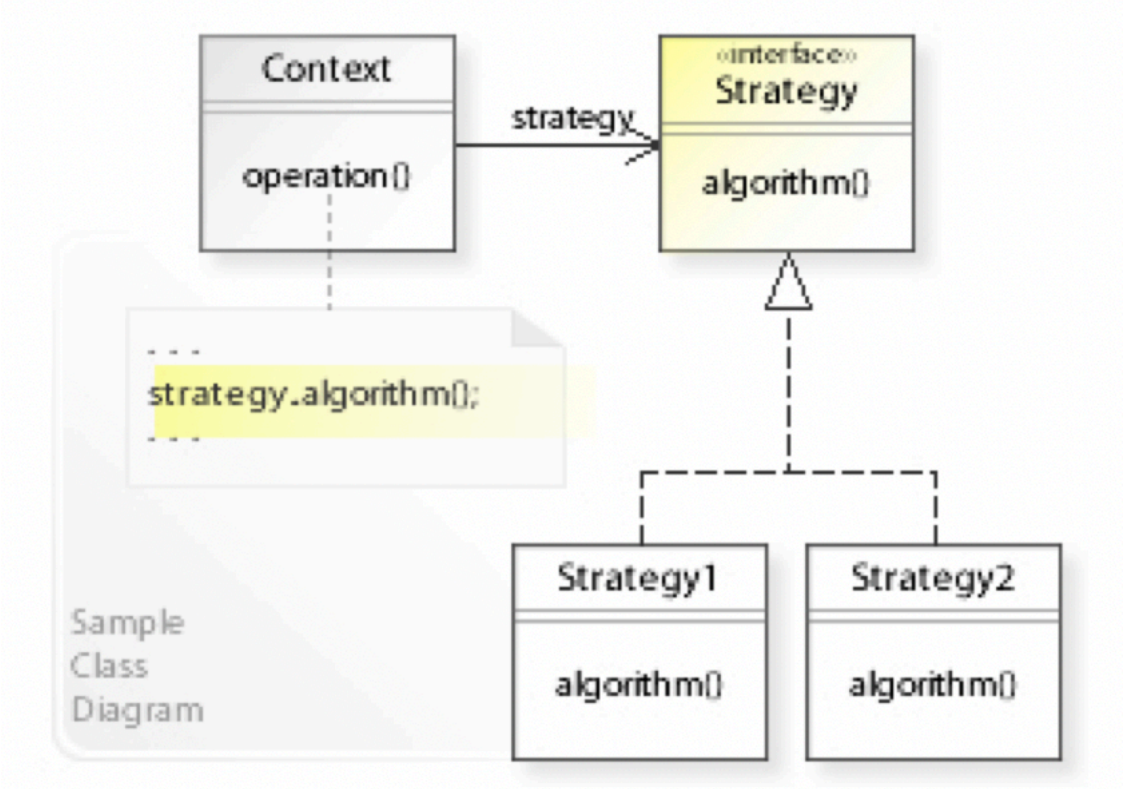


Рисунок 3.1

Алгоритм розроблений для фільтрування:

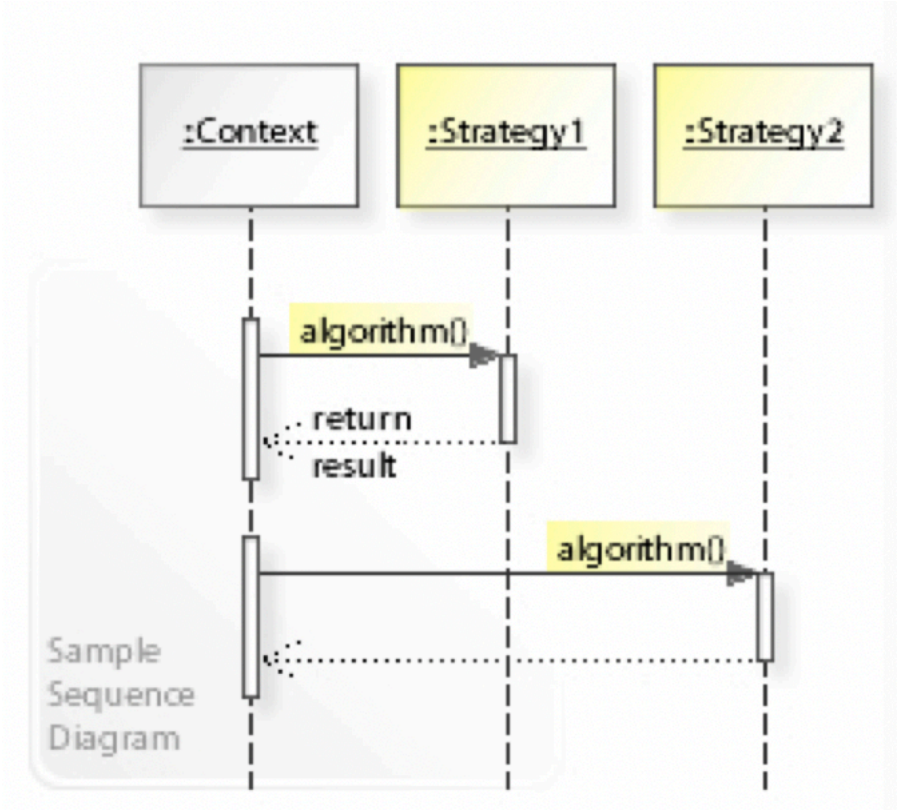


Рисунок 3.2

Інформаційна система містить новини, події та проєкти, кожна новина може бути незалежною або закріплена за подією або проєктом. Подія може містити безліч новин. Новина містить тег (категорію), дату публікації і автора. Подія з часом змінює свій статус (запланована, пройшла), має два варіанти. Порядок новин на головній сторінці можна редагувати, обирати певні. До того ж, реалізована можливість благодійного внеску на зацікавлену тему події або проєкт. Інформаційна система має декілька варіантів мови. Благодійність відбувається для певних проєктів.

3.2. Генерація стратегії системи

Програмний продукт містить наступні складові:

1. **Система збереження та обробки даних:**
    a. Введення та зміна даних через сторінку адміністратора
    b. Обробка введених даних в залежності від типу запису
    c. Завантаження бінарних файлів та малюнків

d. Створення записів різних видів
e. Можливість перекладу записів
f. Зміна динамічних складових сторінок
g. Оптимізація та компіляція даних у прийнятному для системи вигляді

2. **Система виведення даних для користувача**
    a. Відображення статичних HTML сторінок з динамічними змінним частинами
    b. Відображення контенту в зручній для користувача мові
    c. Пошук та фільтрація по записам
    d. Оплата послуг через платіжну систему
    e. Інтуїтивно зрозумілі меню та інтерфейс
3. **Згенеровані дані для БД**
    a. Дані змінних полів
    b. Дані записів різних типів
    c. Дані облікових записів адміністраторів
    d. Дані перекладів
    e. Дані про тестові умовні транзакції
    f. Дані координат на мапі
    g. Дані про тестових згенерованих користувачів

Діаграма генерації даних для бази даних у додатку інформаційної системи

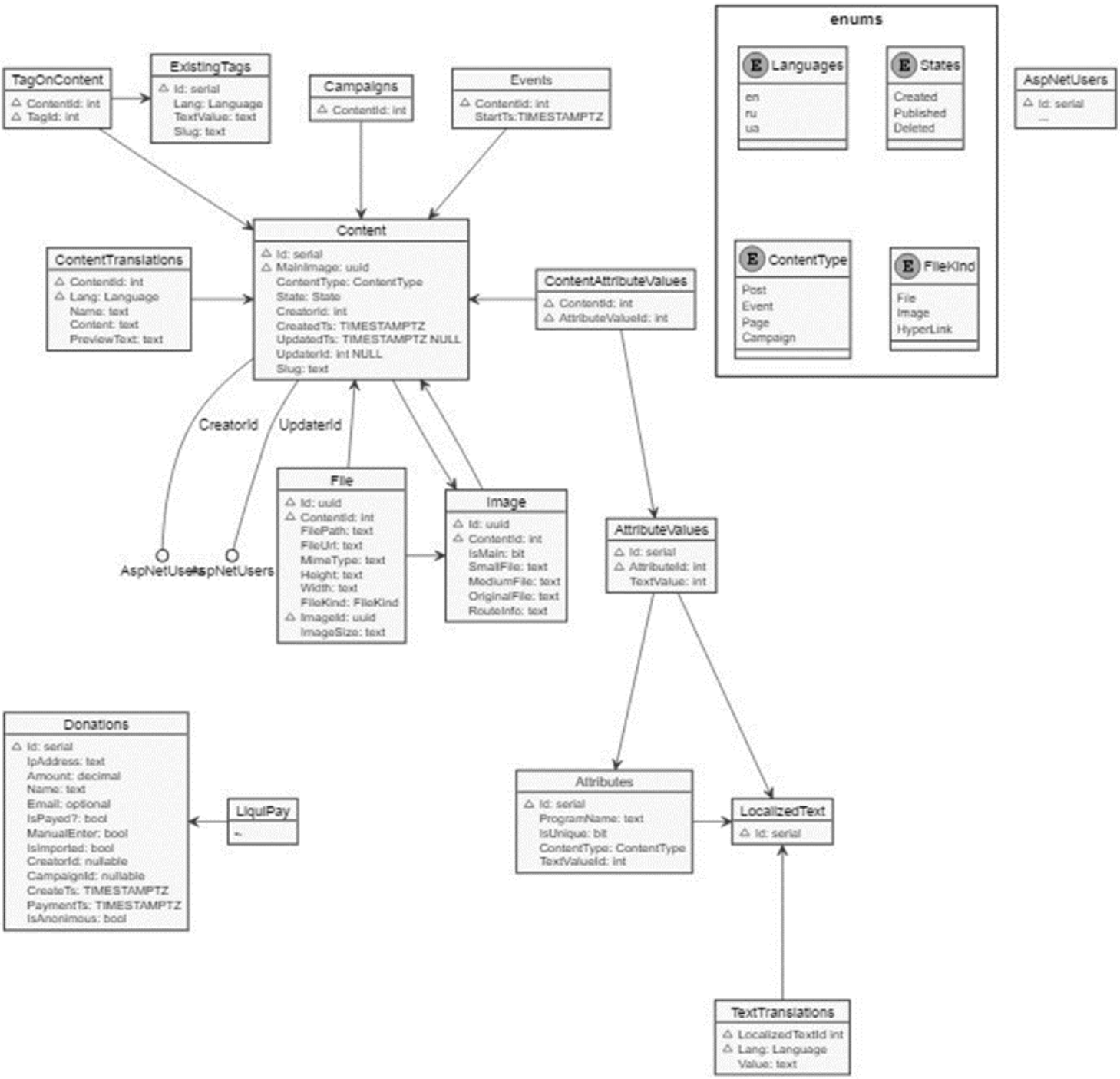


Рисунок 3.3

Рівні доступу до даних:

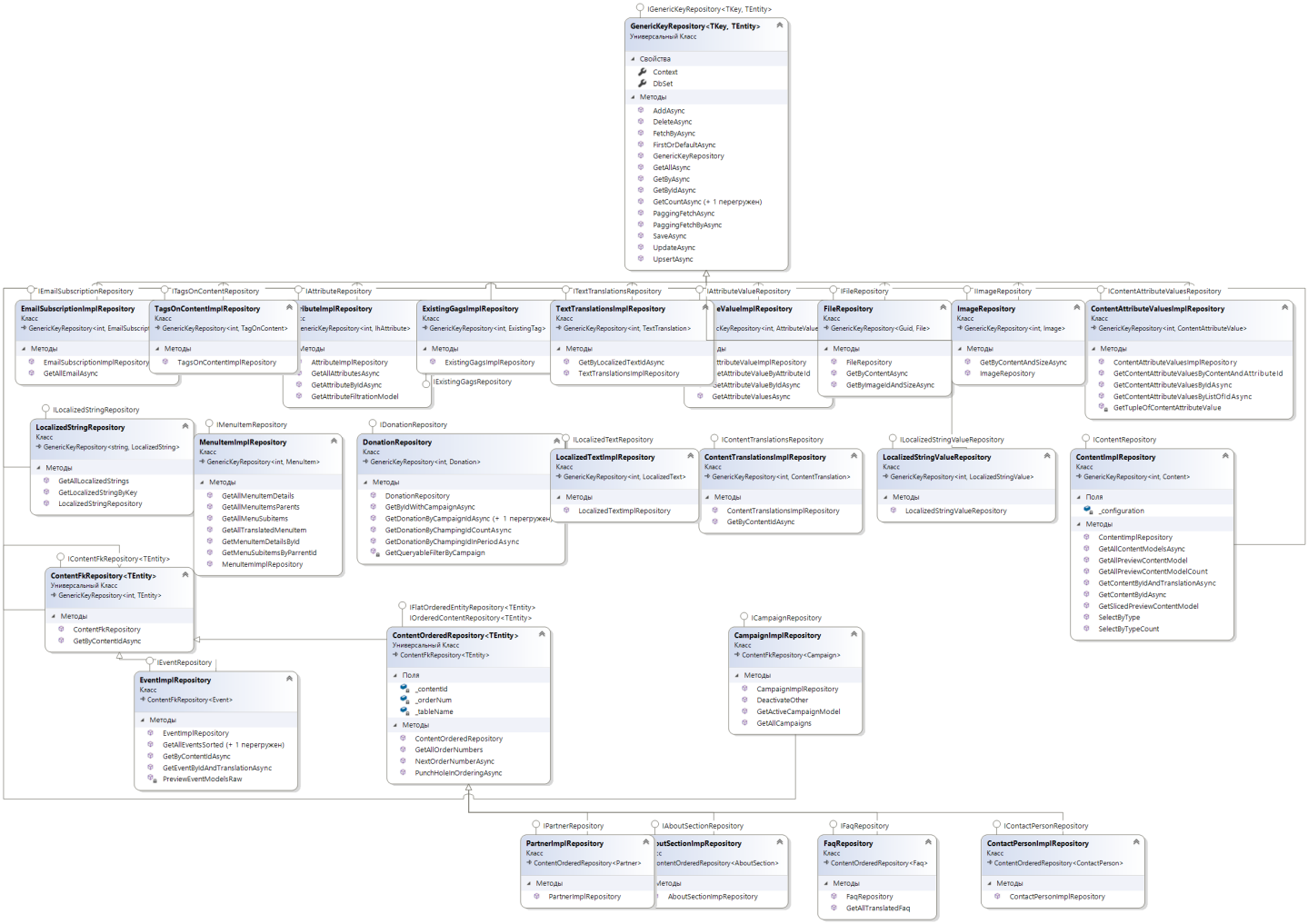


Рисунок 3.4

Опис полів таблиць бази даних:

Таблиця 3.1

Таблиця	**Назва стовпця**	**Тип даних**	**Опис поля**
Categories	Id	integer	Унікальний ідентифікатор
	Value	guid unique	Назва категорії
ErrorType	Id	integer	Унікальний ідентифікатор
	Value	guid unique	Тип помилки
Language	Id	integer	Унікальний ідентифікатор
	Name	string	Мова
MailCategories	MaillingId	integer	Унікальний ідентифікатор листа
	CategoryId	integer	Унікальний ідентифікатор категорії
MailHistories	SubscribeMailId	guid	Унікальний ідентифікатор підписника листа
	Date	timestamp	Дата та час
	Status	statustype	Статус

Продовження таблиці 3.1

	Text	text	Текст листа
	ErrorCode	text	Тип помилки
MailJobs	Id	guid	Унікальний ідентифікатор
	JobId	text	Унікальний ідентифікатор професії
	SubscribeMaillId	guid	Унікальний ідентифікатор підписника за професією
	RetryAttempt	integer	Повторна спроба відправки
Maillings	Id	integer	Унікальний ідентифікатор
	Status	statustype	Статус
MaillingTranslation	Body	string	Текст листа
	Title	string	Заголовок
MailTranslations	MaillingId	integer	Унікальний ідентифікатор листа
	Lang	language	Мова
	Body	text	Текст листа
	Title	text	Заголовок

Продовження таблиці 3.1

MessagesCategories	MessageID	integer	Унікальний ідентифікатор повідомлення
	CategoryID	integer	Унікальний ідентифікатор категорії
StatusType	None	statustype	Пусто
	Success	statustype	Вдале відправлення
	Failed	statustype	Відправлення провалено
	OnDelievery	statustype	В процесі доставки
	Terminated	statustype	Припинено
SubcriberMails	Id	guid	Унікальний ідентифікатор
	UserID	guid	Унікальний ідентифікатор користувача
	MaillingID	integer	Унікальний ідентифікатор пошти користувача
	Status	statustype	Статус

Продовження таблиці 3.1

Subcribers	ID	guid	Унікальний ідентифікатор
	Email	text	Електронна пошта
	Name	text	Повне ім’я
	Lang	language	Мова
SubcribersCategories	SubcriberID	guid	Унікальний ідентифікатор
	CategoryID	varchar (255)	Унікальний ідентифікатор категорії

Генерація обраної стратегії відбувається на основі переданих параметрів та методів як реалізується в моделі MVVM .

BaseStrategy<TEntity>:

- ProcessExtendedItem – витягнення з бази даних спицифічної інформації для даного типу контенту, Event – дата початку події, FAQ – номер в списку (для відображення на фронтенді), Campaign – кількість залучених до проєкту, коштів зібрано або потрібно, ідентифікатор його активності, мапінг у відповідну BLL модель, перетворення в json-object та повернення в даній структурі.

- {
  ResponseType = Succeed
  Data = model
  };
- ProcessExtendedItem – вибір виконуваної процедури на основі переданого параметру ProcessState: created, published, deleted та безпосередній їх виклик з переданими параметрами;
- • CreateAsync – додавання запису в базу даних через механізм ORM-транзакції;
- UpdateAsync – оновлення запису в базу даних через механізм ORM-транзакції;
- DeleteAsync – видалення запису в базу даних через механізм ORM-транзакції;
- IsExisting – перевірка існування в розширеній таблиці запису з інформацією притаманною саме цьому виду контенту, по ключу contentId;

Зазначимо, що всі перелічені методи – віртуальні, тобто їх поведінку можна змінити в наслідуваних від BaseStrategy класах.
OrderContentStrategy<TEntity>(inherits from BaseStrategy<TEntity>):

- SetOrderNumber – зміна порядку відображення контенту, індентифікованого по ключу contentId, серед своєї множини TEntity – певного типу контенту
- CreateAsync – перевизначений метод базового класу з присвоєнням запису певного orderNumber = max{existingOrderNumbers} + 1
- UpdateAsync - перевизначений метод базового класу з доданою логікою впорядкування записів даного типу контенту по orderNumber у випадку оновлення orderNumber даної сутності

Додаткові класи стратегій для кожного типу контенту, що певним чином перевизначають, за необхідністю, методи свого базового класу, наслідують від:

1) OrderContentStrategy – AboutSectionStrategy, ContactPersonStrategy, FaqStrategy, PartnerStrategy
2) BaseStrategy – CampaignContentStrategy, EventStrategy

ContentExtendedStrategyFactory:

- GetStrategyInstance – метод повертає ініціалізовану стратегію, чий вибір базується на переданому параметрі ContentType

Велику роль в додатку грає валідація моделі, тобто перевірка отриманих від клієнта даних на коректність.
BaseValidationStrategy<TEntity>:

- isValid – стандартна перевірка полів (name, text, previewText) контенту на non-empty|null|white-space-only значення
- ValidateModel – десеріалізація json-object до розширеної моделі та її перевірка на специфічні помилки

Спеціалізовані валідаційні стратегії:

- AboutSectionValidationStrategy
- AttributeValidationStrategy
- AttributeValueValidationStrategy
- CampaignValidationStrategy
- ContactPersoneValidationStrategy
- EventValidationStrategy
- FaqValidationStrategy
- PartnerValidationStrategy
- PostValidationStrategy

3.3. Випробування інформаційної системи

При тестуванні було застосовано декілька підходів, було покрито модульним програмуванням 80%.

Для інтеграційних тестів в Android Studio виконувались такі дії:

- Тестовий проект, який використовується для зберігання і виконання тестів.
- Тестовий проект створює тестовий веб-вузол для ТЗ і використовує клієнт тестового сервера для обробки запитів і відповідей з допомогою ТС.
- Засіб запуску тестів використовується для виконання тестів і передачі результатів тестів.

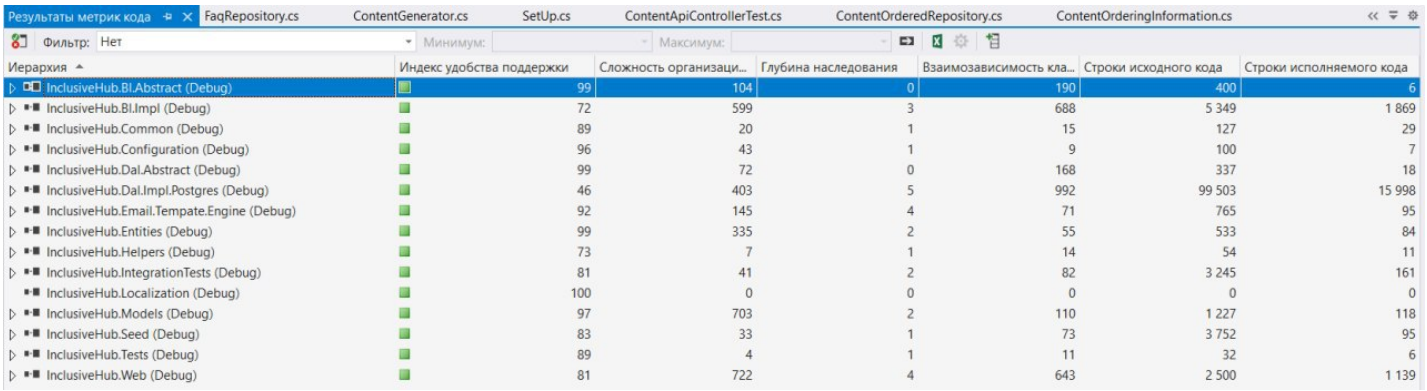
Інтеграційні тести дотримуються послідовності подій зі звичайних кроків тесту Підготовка, Виконання і Перевірка.

- Налаштовується ТС.
- Створюється клієнт тестового сервера для відправки запитів до додатка.
- Виконується крок тесту Підготовка: тестове додаток готує запит.
- Виконується крок тесту Виконання: клієнт відправляє запит і отримує відповідь.
- Виконується крок тесту Перевірка: фактичний відповідь визначається як правильний або неправильний на основі очікуваної відповіді.
- Процес триває до тих пір, поки не будуть виконані всі тести.
- Виводяться результати тесту.

В додатку використовуються платформа тестування xUnit і бібліотека засоби синтаксичного аналізу AngleSharp, тому він також посилається на:

- xunit
- xunit.runner.visualstudio
- AngleSharp

Інспекція до коду представлена на рисунку нижче.



Иерархия	Индекс удобства поддержки	Сложность организаци...	Глубина наследования	Взаимозависимость кла...	Строки исходного кода	Строки исполняемого кода
InclusiveHub.BI.Abstract (Debug)	99	104	0	190	400	6
InclusiveHub.BI.Impl (Debug)	72	599	3	688	5 349	1 869
InclusiveHub.Common (Debug)	89	20	1	15	127	29
InclusiveHub.Configuration (Debug)	96	43	1	9	100	7
InclusiveHub.Dal.Abstract (Debug)	99	72	0	168	337	18
InclusiveHub.Dal.Impl.Postgres (Debug)	46	403	5	992	99 503	15 998
InclusiveHub.Email.Tempate.Engine (Debug)	92	145	4	71	765	95
InclusiveHub.Entities (Debug)	99	335	2	55	533	84
InclusiveHub.Helpers (Debug)	73	7	1	14	54	11
InclusiveHub.IntegrationTests (Debug)	81	41	2	82	3 245	161
InclusiveHub.Localization (Debug)	100	0	0	0	0	0
InclusiveHub.Models (Debug)	97	703	2	110	1 227	118
InclusiveHub.Seed (Debug)	83	33	1	73	3 752	95
InclusiveHub.Tests (Debug)	89	4	1	11	32	6
InclusiveHub.Web (Debug)	81	722	4	643	2 500	1 139

Рисунок 3.5

ВИСНОВОК

У рамках дипломної роботи було досягнуто наведені нижче результати.

Було проведено аналіз додатків, платформ на базі Андроїд та відповідну тематику інформаційної системи, розкрито основні характеристки, архітектуру та класифікацію.

Розроблено алгоритм, що допомагає генерувати стратегію реалізіції додатку, а саме, фільтрування проєктів, подій та новин.

Додаток був протестований і показав свою працездатність, як на стандартних емуляторах, взятих із SDK Android, так і на реальних пристроях на платформі Android (планшетному ПК та смартфоні).

Розділ «Аналіз інформаційних систем на базі Android» описує загальні положення Android систем, платформи мобільних додатків, а також класифікацію, архітектуру та характеристику платформ.

Розділ «Структура. Android додатків» розкриває в собі структуру додатків під платформу Android, компоненти Android додатків, яка активувати компоненти та маніфест файли. Детально розкриває використання ресурсів та їх керування в додатках Android.

Розділ «Практична реалізація розширених можливостей» містить в собі алгоритм для фільтрування проєктів, новин та подій, детальну генерацію стратегії системи та випробування інформаційної системи.

ПЕРЕЛІК ПОСИЛАНЬ

1. Інструменти Visual Studio для Xamarin // Microsoft URL: https://visualstudio.microsoft.com/ru/xamarin/?rr=https%3A%2F%2Fwww.goo gl e.com%2F
2. . Казакова А. Н. Дослідження способів зберігання інформації в додатках Android // Матеріали конференцій ДНДІ "Нацрозвиток". Грудень 2017. М.: ГНИИ "Нацразвитие", 2018.
3. Які мови програмування потрібно знати, щоб розробляти програми під Android?// Medium URL: https://medium.com/nuances-ofprogramming/к
4. Ким В. Ю. Особливості розробки дизайну інтерфейсу для мобільного додатка // Нові інформаційні технології в автоматизованих системах. 2015 року.
5. ASP.NET Core[Електронний ресурс] // Режим доступу: https://docs.microsoft.com/ru–ru/aspnet/core/?view=aspnetcore–3.1/
6. RestApi [Електронний ресурс] // Режим доступу: https://restfulapi.net/
7. Angular [Електронний ресурс] // Режим доступу: https://angular.io/
8. PostgreSQL [Електронний ресурс] // Режим доступу: https://www.postgresql.org/about/
9. Клієнт–серверна архітектура MVC [Електронний ресурс] // Режим доступу: https://docs.identityserver.io/en/latest/quickstarts/2_interactive_aspnetcore.html
10. Адам Фримен ASP.NET MVC 5 с примерами на C# 5.0 для профессионалов, 5–е издание = Pro ASP.NET MVC 5. — М.: «Вильямс», 2014. — 736 с. — ISBN 978–5–8459–1911–3.

11. Джесс Чедвик, Тодд Снайдер, Хришикеш Панда ASP.NET MVC 4: разработка реальных веб–приложений с помощью ASP.NET MVC = Programming ASP.NET MVC 4: Developing Real–World Web Applications with ASP.NET MVC. — М.: «Вильямс», 2013. — 432 с. — ISBN 978–5–8459–1841–3

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

ДИПЛОМНИЙ ПРОЄКТ

на тему: «*Розробка інформаційної системи на базі Android з розширеними можливостями та покращеною адаптивнісю*»

Виконав ст. гр . ІСДМ-61 Дмитрій КАШИЧ

Керівник ДП: доктор технічних наук, професор Андрій БОНДАРЧУК

Київ - 2022

Призначення розробки

Цілі розробки

Призначенням розробки інформаційної системи

Цілями розробки є:

- спрощення процесу інформування зацікавлених користувачів про події та спеціальні програми
- спрощення процесу пошуку та приєднання до спеціальних програм
- збільшення аудиторії небайдужих до певної проблематики за рахунок надання можливості допомогти через сервіс та приєднатися до співпраці

Задачі розробки

- Публікація та перегляд інформації про діяльність організації
- Розміщення та перегляд новин
- Розміщення та перегляд подій, навчальних програм
- Публікація інформації про партнерів організації, запрошення потенційних спонсорів до благодійності, можливість здійснити пожертву
- Перегляд та публікація кампаній зі збору коштів.
- Активна взаємодія з користувачем: можливість замовити зворотній дзвінок, оформити запит на проведення власної події, підписка на новини та події
- Можливість зміни мови застосування

Огляд існуючих аналогів

- https://promprylad.ua/ua/:

- відсутність подій;

- відсутній пошук по сайту;

- відсутність адаптивності на декілька мов

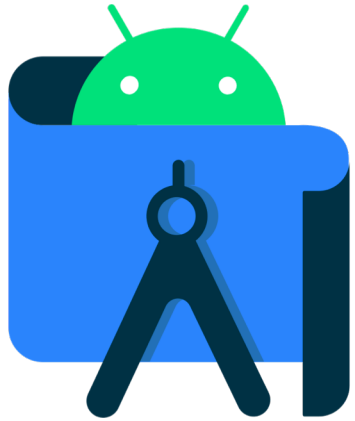
- https://www.warm.if.ua/about_us

- відсутність адаптивності на декілька мов;

- відсутність змоги реєструватися на майбутні події

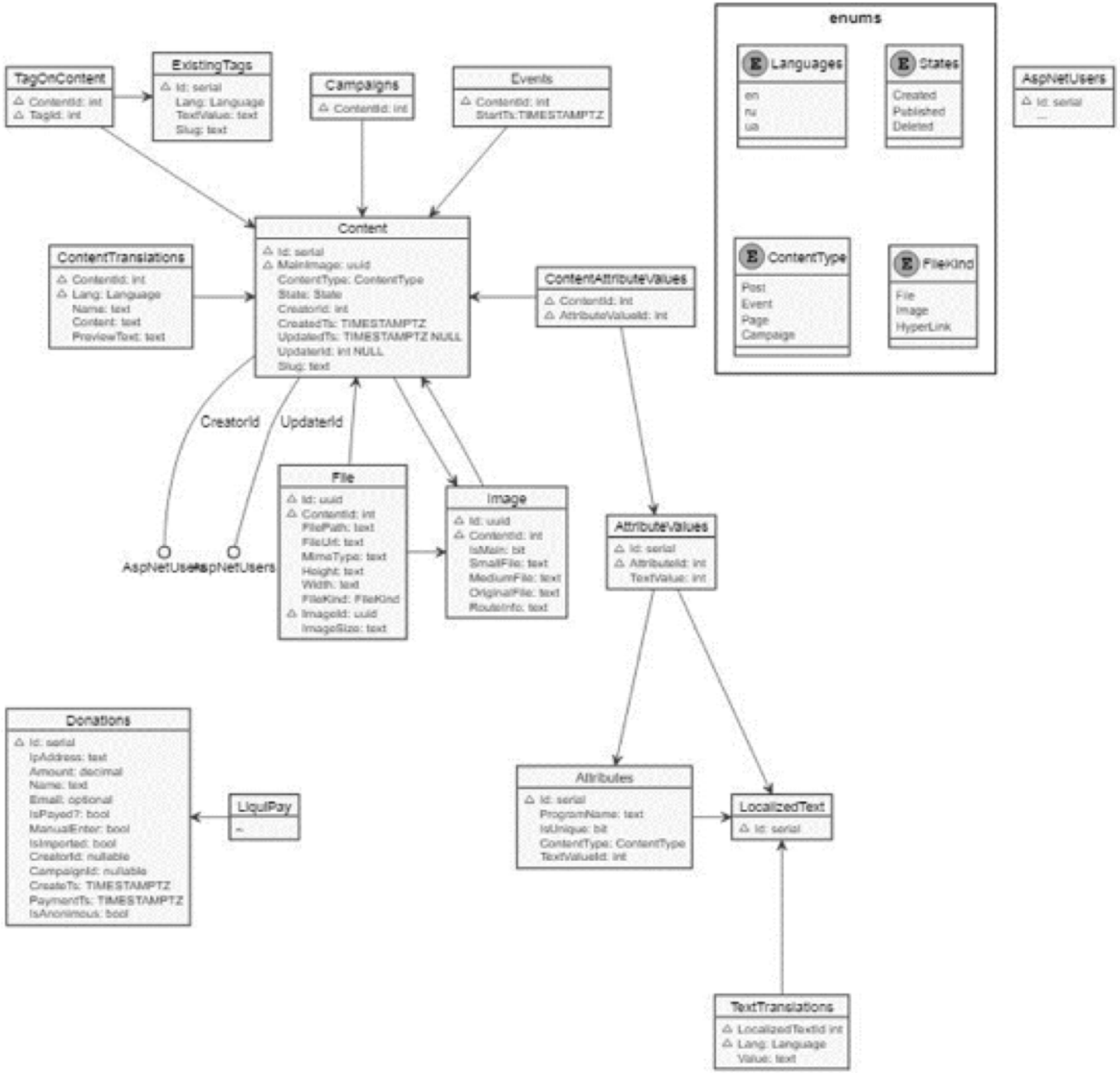
Середовище розробки та мова програмування

android studio

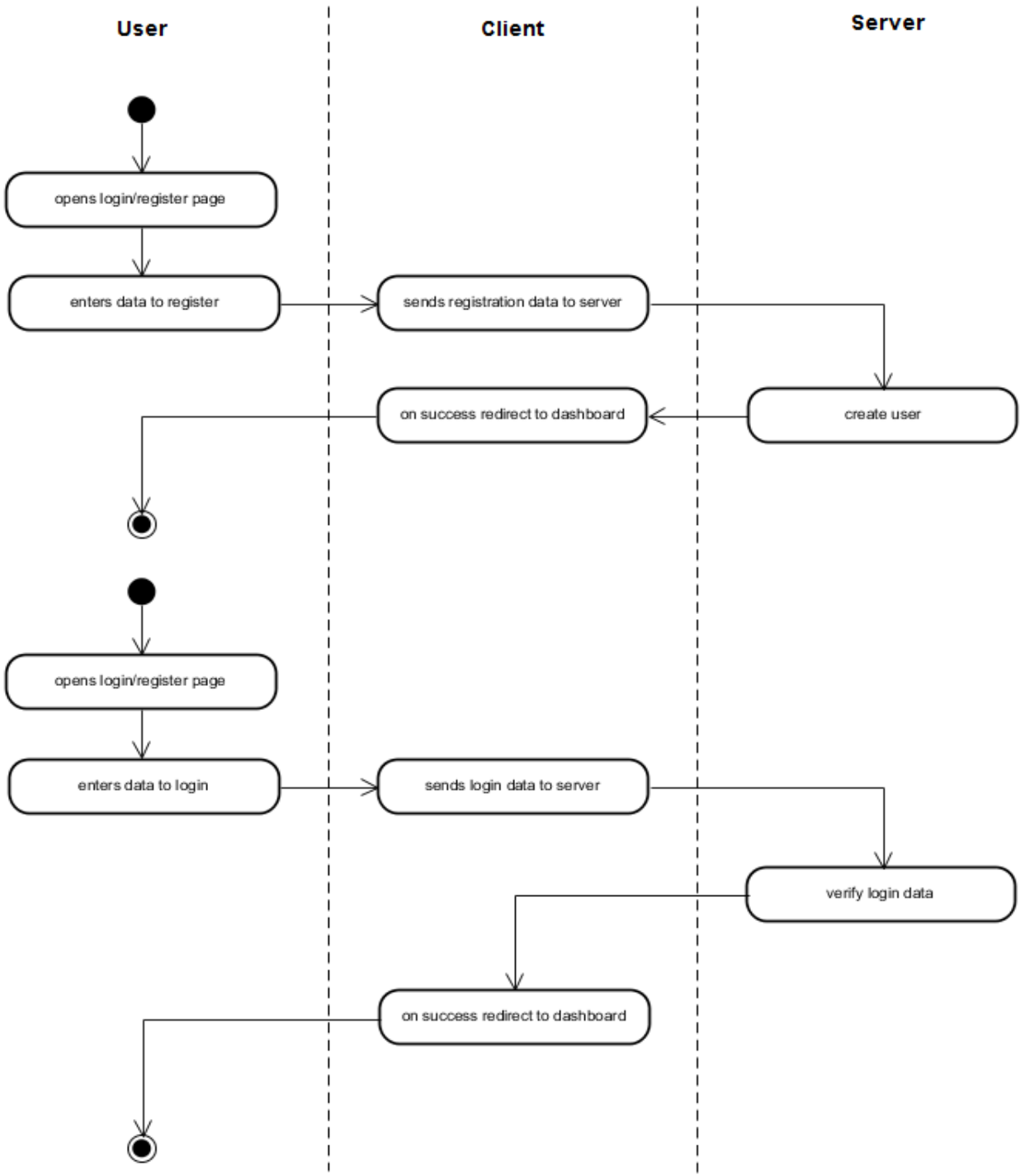


Kotlin

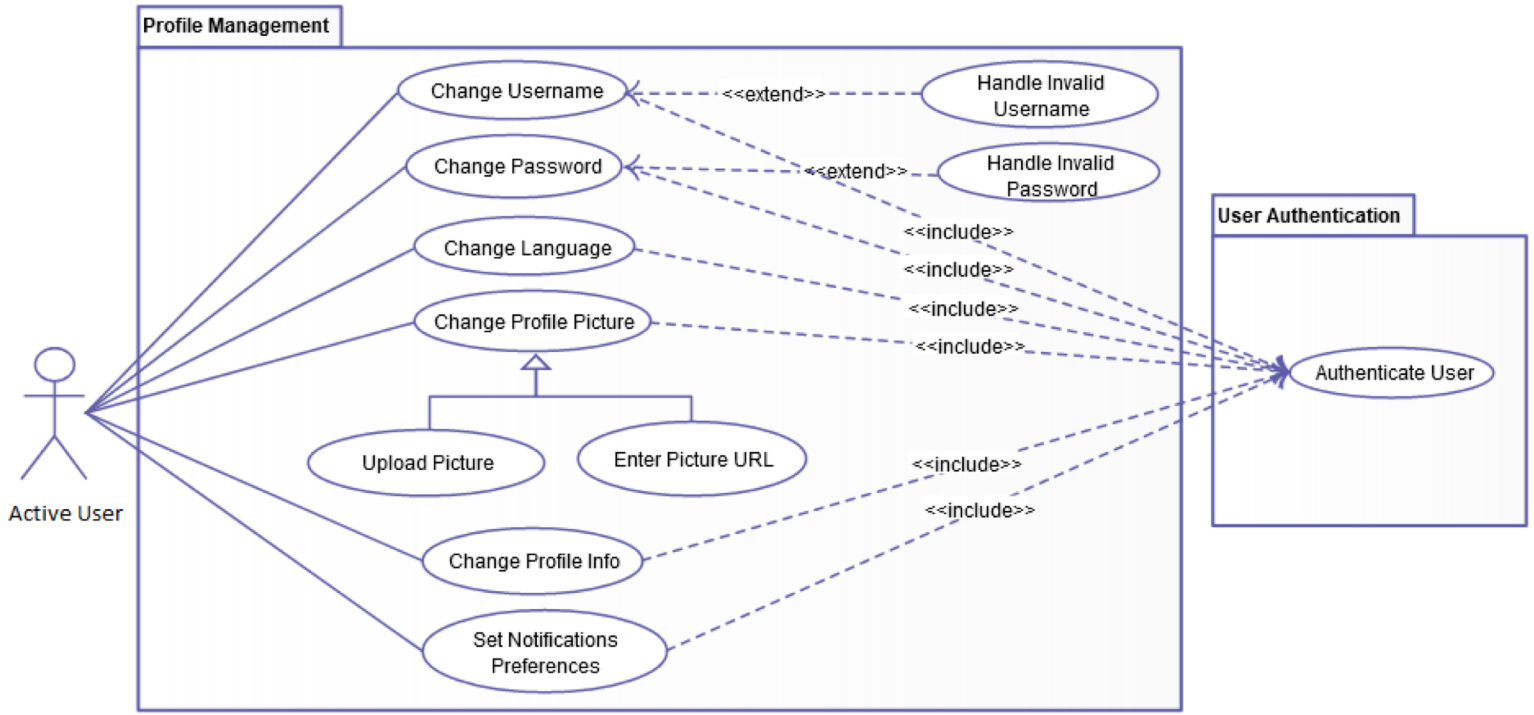
Опис функціональної моделі бази даних



Опис функціональної моделі



Опис функціональної моделі



Дякую за увагу!

Виконав ст. гр . ІСДМ-61 Дмитрій КАШИЧ

Керівник ДП: доктор технічних наук, професор Андрій БОНДАРЧУК