Форма № Н-9.02

Житомирський державний університет імені Івана Франка

Фізико-математичний факультет

Кафедра прикладної математики та інформатики

Пояснювальна записка

до дипломного проекту (роботи)

Магістра

на тему:

«Розробка електронного посібника для платформи Android»

Виконав:

студент 6 курсу 63 групи

напряму підготовки

8.04030201 «Інформатика*»

Кухтюк В.О.

Керівник: доктор педагогічних наук Спірін О.М.

Рецензент: кандидат технічних наук Морозов А.В.

Житомир - 2015

План

ВСТУП

Android – операційна система для смартфонів, планшетів і нетбуків, що базується на модифікованому ядрі Linux. Вона була створена компанією Google, яка ефективно співпрацювала з учасниками Open Headset Alliance. Ця новітня операційна система має велику спільноту розробників, які розширюють її функціональність.

Програмування під Android, завдяки гнучкості платформи, дозволяє створювати корисні програми для будь-яких потреб. Оскільки сьогодні переважна більшість смартфонів працюють саме під управлінням Android, то розробка під цю систему відкриває багато перспектив для управління бізнесом та надання послуг для великої аудиторії користувачів, які кожен день завантажують нові програми для своїх пристроїв на Google Play.

Актуальність. Розвиток мобільних пристроїв не стоїть на місці. Вони набувають все більшої популярності серед користувачів. А домінуючою на ринку мобільною платформою є операційна система Android. Вона є найпопулярнішою мобільною операційною системою в світі, що є добрим приводом для вивчення як самої системи, так і особливостей розробки програм для неї.

Актуальною буде і розробка електронних посібників для цієї системи, оскільки звичайні підручники вже не користуються такою популярністю, як раніше. Електронний посібник достатньо просто встановити на пристрій з ОС Android в той час, коли по звичайний потрібно йти в бібліотеку або в книжний магазин. Крім того, в мобільному пристрої може міститися десятки, а то й сотні електронних посібників. Інколи їх може вміститися навіть більше, чим є звичайних підручників в самій бібліотеці. Це залежить від того, скільки пам’яті є на пристрої. Люди не мають тепер носити з собою улюблені підручники, адже все це вони можуть мати на своєму Android-пристрої. Це являється більш зручним способом набуття знань.

Предмет дослідження:Електронний посібник для ОС Android.

Об’єкт дослідження: Розробка в програмному середовищі електронного посібника для платформи Android

Мета роботи. Проаналізувати етапи розвитку версій операційної системи Android, дослідити архітектуру її програм, середовище розробки програм для неї, а також розробити проект під назвою «Електронний посібник».

Новизна роботи. Застосування сучасної технології програмування для ОС Android, а саме мови Java, якою буде створюватися програма в середовищі Android Studio.

Завдання на дипломну роботу:

1. Здійснити пошук інформації по тематиці дипломної роботи та провести її аналіз та дослідження;
2. Ознайомитись з різними версіями ОС Android.
3. Розглянути структуру програми для мобільної платформи;
4. Розглянути сучасні програмні засоби для розробки під операційну систему Android;
5. За допомогою програмних засобів розробити проект, що буде мати назву «Електронний посібник».

РОЗДІЛ 1. ТЕОРЕТИЧНА ЧАСТИНА.

1.1. Версії операційної системи Android.

Інтерфейс користувача платформи Android максимально адаптований для того, щоб керування пристроєм здійснювалося за допомогою пальців, без використання пера для вводу (стилусу). Версії для цієї операційної системи постійно дороблюються та оновлюються, що робить її перспективною. Цікавим є факт, що кожна нова версія Android отримує цікаву назву десерту в алфавітному порядку. Спочатку компанія хотіла використовувати імена відомих роботів, але відмовилася через проблеми з авторськими правами. Перші дві версії, 1.0 Astroboy та 1.2 Bender були названі іменами саме роботів, але потім були перейменовані в назви десертів [5].

Версіі ОС Android:

1) **Android 1.0 G1** (*Рис 1.1*)**,** що отримала кодову назву Apple Pie (яблучний пиріг), є базовою версією, яка побачила світ у вересні 2008 року. В її основі було покладено ядро Linux 2.6.25, а також підтримка файлової системи FAT32. Поява цієї версії викликала до себе великий інтерес, бо це означало, що такий гігант, як Google, виходить на ринок мобільних платформ. Це версія включає в себе базовий набір функцій: GPS, Bluetooth, Google Maps (разом із Street View) та стандартні програми такі, як Календар, Контакти, YouTube та ін. При запуску версії G1 з'являється магазин Android Market, який нараховував 35 програм. Протягом трьох років, це число зросло до 250 000 [1].

Рис 1.1- логотип Android 1.0

2) **Android 1.2 Banana Bread** (*Рис 1.2*) – версія, що була випущена тільки для T-Mobile G1 (HTC Dream) в лютому 2009 року. Android 1.2 «Banana Bread» (банановий хліб) включала в себе виправлення деяких проблем, зміни в API. Були додані деталі та відгуки до карт, збільшений період відключення екрану при використанні в режимі

Рис 1.2 – логотип Android 1.2

телефону. Також додані кнопки «Show» та «Hide» в меню виклику та підтримка збереження вкладів із MMS [3].

3) **Android 1.5 Cupcake** (*Рис 1.3*) з'явилася в квітні 2009 року. В цій версії врахували помилки попередніх та виправили недоліки. Ця версія стала стабільнішою та швидшою за G1. Також було введено чимало нових корисних функцій. Серед них такі, як: пошук по категоріям з використанням фільтру, програмна клавіатура з функцією автозаповнення, а також підтримка акселероментра. Камера отримала можливості переходу між фото і відео режимами, а також вбудовану фотогалерею, запис та відтворення відео в 3GP та MPEG-4 форматах, видалення декількох вибраних фотографій. Також став доступним сервіс публікації фото та відео в Інтернеті. Користувачі були раді новим віджетам на робочому столі та анімації при переключенні вікон. Cupcake (кекс) настільки вразив ринок мобільних систем, що такі гіганти, як Samsung, HTC та LG підписали договори з Google про співпрацю, що послугувало великим поштовхом для розвитку цих компаній, бо вони на той час не могли похвалитися значним ростом у сфері мобільних пристроїв[1].

Рис 1.3 – логотип Android 1.5

4) **Android 1.6 Donut** (*Рис 1.4*) компанія Google випустила у вересні 2009 року. З виходом Donut (пончик) стався бум Інтернет-програм для Android. Оновився Android Market, який дозволяв користувачам простіше знаходити програми, для кожної з яких з'явилися скріншоти. Також став доступним пошук в Інтернеті, історії, закладках, контактах прямо з головного екрана, реалізована функція багатомовного голосового пошуку. Операційна система стала більш інтегрованою з фото та відеокамерою, галерея дозволяла вибрати декілька об'єктів для видалення. З'явилася можливість відслідкувати програму або сервіс, які споживають найбільше заряду батареї. В цій версії

Рис 1.4 – логотип Android 1.6

ще з’явилася підтримка сотового стандарту CDMA, а також декількох дисплеїв з більшою роздільною здатністю: QVGA та WVGA[2].

5) **Android 2.0, 2.1 Enclair** (*Рис 1.5*), що вийшла в жовтні 2009, стала першою версією операційної системи, яка мала можливість використання декількох аккаунтів Google. В Android 2.0 Éclair (тістечко) користувачі були раді оптимізації швидкості роботи, оновленому інтерфейсу та покращеної компоновки віртуальної клавіатури для підвищення швидкості набору й зменшення помилок при введенні. До камери додали спалахування, цифровий зум та макрозйомку. Також з’явилася можливість пошуку в збережених SMS та MMS, підтримка HTML5, Bluetooth 2.1 і Microsoft Exchange. У версію 2.1 були додані «живі» шпалери та покращення контрастності фону. Тоді Android показав вражаюче зростання встановлень і тому отримав перспективу роботи на більш потужних пристроях, таких як планшети та нетбуки [4].

Рис 1.5 – логотип Android 2.0, 2.1

6) **Android 2.2 Froyo** (*Рис 1.6*) вийшла в травні 2010. Як і можна було сподіватися, Froyo (заморожений йогурт) приніс значні покращення та доповнення. Серед них такі, як швидша робота системи, що зросла приблизно в 3-5 раз. Таких показників вдалося досягти за рахунок використання Dalvik Virtual Machine Juit-In-Time Compiler. Це була перша версія операційної системи, що отримала підтримку Adobe Flash 10.1 і це дозволяло системі скоріше завантажувати сторінки з великим вмістом скриптів JavaScript. З’явилася функція автооновлення програм та можливість встановлення їх прямо на карту пам’яті, що дозволило значно економити вбудовану пам’ять пристрою. Крім того, Google реалізував повноцінну роботу з мережами Wi-Fi і навіть дав можливість смартфону стати точкою підключення до мережі через Bluetooth. До цього часу Android вже затвердився на ринку мобільних платформ і став конкурентом для найбільш популярної операційної системи компанії Apple, iOS [2].

Рис 1.6 – логотип Android 2.2

7) **Android 2.3 Gingerbread** (*Рис 1.7*) була випущена в грудні 2010 року. Разом з її виходом був представлений набір засобів для розробки – SDK та NDK. Поява Gingerbread (імбирне печиво) стала проривом у світі програмного забезпечення для мобільних пристроїв. Деяких змін зазнав інтерфейс користувача, який був перероблений з метою спрощення користування смартфоном. За рахунок цього збільшилась енергоефективність системи. Була покращена і стандартна клавіатура та її швидкодія. Зазнав змін також екран, котрий тепер мав підтримку надвисоких роздільних здатностей, таких як WXGA і вище. Були введені нові формати відео WebM, VP8, а також стандарт аудіо AAC та нові звукові ефекти. Новинкою стали відео дзвінки на інші телефони (SIP WoIP-телефонія). Був посилений контроль за програмами і живленням, з'явилася вбудована підтримка декількох камер. Вихід Android 2.3 співпав із збільшенням активності Android-розробників, внаслідок чого стався значний ріст Android Market [1].

Рис 1.7 – логотип Android 2.3

8) **Android 3.0, 3.1, 3.2 Honeycomb** (*Рис 1.8*) вийшла в лютому 2011 року. Різниця між Honeycomb (медові соти) та ранніми версіями Android в тому, що вона являється спеціальною версією з абсолютно новим інтерфейсом та присутністю повноцінної оптимізації під пристрої з великими екранами, що призначена для роботи на планшетах (MID, tablets). Ця версія містить в собі «голографічний» інтерфейс GUI з підтримкою 3D і модернізовану багатозадачність. До основних нововведень системи можна віднести:

Рис 1.8 – логотип Android 3.0

• підтримку віртуальних робочих столів, кожний з яких може мати свій набір віджетів та ярликів;

• покращену віртуальну клавіатуру, що дозволяє швидко вводити текст на великому дисплеї;

• можливість підключення периферійних пристроїв, таких як клавіатура, камера, мишка та ін [4].

9) **Android 4.0 Ice Cream Sandwich** (*Рис 1.9*), представлена в жовтні 2011, призвела до справжнього фурору. Ціллю розробників було з'єднати смартфонні та планшетні версії в одну, щоб можна було встановлювати і на смартфони, і на планшети. Ice Cream Sandwich (вафельне морозиво) вражає своєю кількістю оновлень, таких як: розблокування екрану жестами, єдиний інтерфейс для смартфонів та планшетів, стандартна підтримка скріншотів, вбудований редактор фотографій, краща якість роботи камери, зум під час запису відео та заповільнена зйомка, контроль за інтернет-трафіком, функція Android Beam для обміну між двома пристроями, покращена безпека та багато інших нових оновлень, якими порадувала користувачів нова версія Android [3].

Рис 1.9 – логотип Android 4.0

10) **Android 4.1, 4.2, 4.3 Jelly Bean** (*Рис 1.10*) з'явилася в червні 2012 року. Jelly Bean (желейні цукерки) стала найбільш швидкою та функціональною версією. Була збільшена швидкість завантаження інтерфейсу, покращений пошук. Покращення швидкодії інтерфейсу було зумовлено оптимізацією алгоритму обробки графіки. При дотику до екрану, центральний процесор задіює додаткову потужність для завантаження інтерфейсу. В новій версії також додали нове меню повідомлень, де не тільки можна відслідковувати активність смартфону, а й керувати різними функціями. З випадаючого меню можна відповідати на повідомлення, передзвонити контакту, від якого був пропущений виклик. У версії 4.2 було додано підтримку Wireles display, завдяки чому можна транслювати фільми чи зображення на зовнішній дисплей. Також з'явилася розумна клавіатура з можливістю друкування без відриву від екрану (Swype-клавіатура), за допомогою якої можна вводити

Рис 1.10 – логотип Android 4.1, 4.2, 4.3

текст, не відриваючи палець від екрану. Користувач проводить по буквах, а клавіатура підказує потрібне слово [1].

11) **Android 4.4 KitKat** (*Рис 1.11*) вийшла в жовтні 2013. Серед головних змін можна виділити інтерфейс окремих системних програм і елементів. В новій версії у програмі Телефон з'явилися великі кнопки для виклику абонента та можливість пошуку потрібного абонента, не покидаючи програму для здійснення дзвінків. Також серед важливих оновлень є і підтримка застарілих телефонів. В KitKat серйозно опрацьований механізм оптимізації пам'яті, що дозволяє задіяти на 16% менше ОЗУ, чим Jelly Bean. В результаті смартфони базового рівня з оперативною пам'яттю 512Мб можуть запускати програми, які раніше потребували значний об'єм ОЗУ [5].

Рис 1.11 – логотип Android 4.4

12) **Android 5.0 Lollipop** (*Рис 1.12*) вийшла в листопаді 2014 року. В ній приміняється тільки середовище ART (Android Runtime Environment). Перевага ART перед Dalvik полягає у швидкості завантаження програм. Також окремої уваги заслуговує енергозберігаючий режим, який вже давно містився у фірмових прошивках сторонніх розробників. Тепер режим економії заряду доступний всім пристроям. Слід зазначити, що змінилася і тема всього інтерфейсу з чорного кольору на білий. Зазнала змін також клавіатура, на якій шрифт став краще читатися. Ще одна особливість, яка була в повній мірі реалізована в Lollipop (льодяник) – це перехід між користувачами. Якщо користувач коли-небудь працював з системою користувачів Windows, то не знайде нічого нового. Кожен користувач може бути захищений паролем, а внутрішні папки (музика, фото) можуть бути закриті для загального доступу. Зазнала змін також програма моніторингу заряду батареї, яка тепер вміє передбачати скільки часу пристрій буде працювати без підзарядки. Lollipop приніс дуже багато інших

Рис 1.12 – логотип Android 5.0

оновлень, які складно перерахувати. Це – логічне і конструктивне продовження всієї лінійки операційних систем Android.

Компанія Google може пишатися тим, що її операційна система встановлена на більше, чим 1 млрд. мобільних гаджетах в світі. А за декілька років було випущено більше 11 тис. різних моделей Android-пристроїв [2].

1.2 Розробка програм для Android.

Android – це безкоштовна операційна система, яка базується на Linux з інтерфейсом програмування Java. Вона підтримує 2D та 3D-графіку, використовуючи бібліотеки OpenGL, а також зберігання даних в базі SQLite. Кожна програма, розроблена на Android, запускається у власному процесі і тому вона ізольована від інших запущених програм, а це не дає неправильно працюючій програмі безперешкодно завдати шкоди іншим.

Пакет засобів для розробки програм на Android – Software Development Kit (SDK Android) містить всі необхідні для створення, компіляції та запакування Android-програм. Ці програми, в основному розробляються із застосуванням мови програмування Java. Android SDK містить в собі Android Debug Bridge, який є інструментом, що дозволяє підключитися до віртуального або реального пристрою з метою керування ним чи налагодження програми [6].

Компанія Google пропонує два інтегрованих середовищ розробки (Integrated Development Environment) для розробки нових програм:

• IDE Android Studio, яке розроблене компанією Google. Це середовище базується на IntelliJ IDE.

• The Android Developer Tools (ADT) базується на Eclipse IDE. ADT – це набір компонентів (плагінів), які розширяють можливості Eclipse IDE для розробки під операційну систему Android.

Обидва середовища містять всю необхідну функціональність для створення, компіляції, налагодження та розгортання Android-програм. Вони також дозволяють розробнику створювати і запускати віртуальні пристрої Android для тестування [7].

1.2.1 Архітектура Android-програми.

Архітектура програм на Android заснована на ідеї багатократного використання основних блоків для будови, що називаються компонентами. Операційна система побудована так, що будь-яка програма може запускати необхідний компонент іншої програми. Коли система запускає компонент, то вона запускає процес програми, якій він належить та створює необхідні для нього екземпляри класів. Тому в ОС Android, на відміну від більшості інших систем, немає єдиної точки входу. Оскільки кожна програма запускається в окремому процесі та є обмеження на доступ до файлів, то програма не може безпосередньо активувати компонент іншої програми. Таким чином, для його активації необхідно відправити системі повідомлення про намір запустити певний компонент і система запустить його [6].

Загалом, Android-програма складається з:

• Java-класів, які є підкласами основний класів із Android SDK (View, Activity, ContentProvider, Service, BroadcastReciever, Intent) та Java-класів, в яких нема батьківських в Android SDK.

• Маніфесту програми, ресурсів (рядків, зображень і т.д.) та файлів.

Основні компоненти програм для Android (*Рис 1.13*):

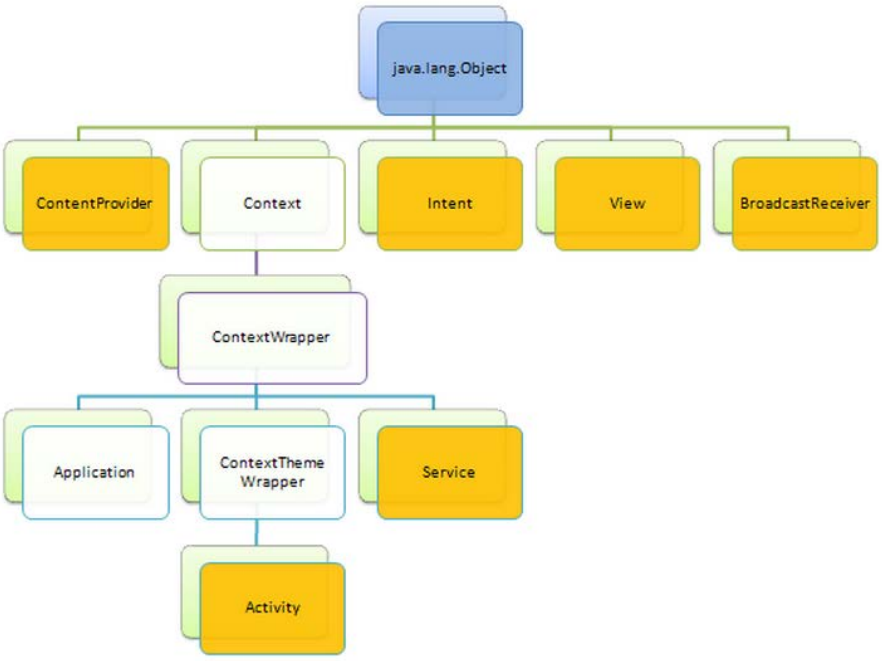

Рис 1.13 – Ієрархія основних компонентів Android-програми

• **Activity** – це видима частина програми і представляє собою екран інтерфейсу користувача, який представлений класом Activity та розміткою у вигляді XML-файлу. Програма на Android може складатися з декількох Activity та може переключатися між ними під час роботи. Наприклад, якщо є програма, призначена для роботи з соціальною мережею, то тут перша Activity може слугувати для відображення власного профілю, друга – для відображення списку друзів, третя – для перегляду повідомлень. Не дивлячись на те, що для користувача програма постає єдиним цілим, всі Activity не залежать одна від одної. В зв'язку з цим, будь-яка з них може бути запущена з іншої програми, яка має доступ до Activity даної програми. Наприклад, програма камери може запустити Activity, яка відображає список друзів користувача, щоб переслати комусь із них тільки що зроблену фотографію [8].

Activity також має свій життєвий цикл, на протязі якого вона знаходиться в одному із трьох станів:

1) Активно і виконується – коли інтерфейс користувача знаходиться на передньому плані, видимий і має фокус. Код програми виконується тільки при таких умовах.

2) Призупинено – коли інтерфейс користувача втратив фокус, але все ще видимий. В такому стані код не виконується.

3) Завершено – коли інтерфейс користувача невидимий. Як і в призупиненому стані, код не виконується. Нема гарантій, що об'єкт Activity та пов'язані з ним інші об'єкти знаходяться в пам'яті, коли Activity знаходиться в цьому стані або в призупиненому [11].

• **Services** – служби, які працюють у фоновому режимі, виконуючи різні задачі. Вони не мають інтерфейсу користувача, але можуть повідомляти користувача через систему повідомлень Android. Цим сервіси і відрізняються від Activity, які виконуються тільки тоді, коли їх інтерфейс користувача знаходиться на передньому плані. До сервісів можна віднести програвання музики у фоновому режимі, поки користувач використовує іншу програму.

Сервіс може бути запущений іншим компонентом, наприклад, Activity та після цього може працювати самостійно або бути зв’язаним з цим компонентом і взаємодіяти з ним.

• **Content Providers** (контент-провайдери) служать для управління даними програми. Дані можуть зберігатися в базі SQLite, в файловій системі, в мережі або в будь-якому іншому доступному для програми місті. Контент-провайдер дозволяє іншим програмам, якщо у них є відповідні права, робити запити або міняти дані. В системі Android є такий контент-провайдер, який управляє інформацією про контакти користувача. Тому, будь-яка програма, що має права на доступ до цих даних, може зробити запит на читання і запис інформації якого-небудь контакту. Контент-провайдер може також використовуватися для читання і запису приватних даних програми, не призначених для доступу зовні [8].

• **Broadcast Receiver** (Приймач широкомовних сповіщень) – компонент, що приймає широкомовні сповіщення. Ці сповіщення зазвичай створюються самою системою, наприклад, повідомлення про те, що екран відключився або низький заряд акумулятора. Можуть інсценувати такі сповіщення і програми, наприклад, програма розсилає повідомлення іншим програмам про те, що деякі дані завантажені та доступні для використання.

Приймач хоч і не відображає інтерфейсу, але може створювати повідомлення на панелі станів, щоб сповістити користувача про появу широкомовного сповіщення. Такі приймачі служать провідниками до інших компонентів та призначені для виконання невеликого обсягу робіт, наприклад, запустити відповідний події сервіс. Програма може реєструватися як приймач певних подій та може бути запущена, якщо така подія відбудеться [9].

• **Intent** – клас, об’єкти якого служать для передачі повідомлень, які називаються намірами, між основними компонентами програм. Наміри – це асинхронні повідомлення, які дозволяють програмі запросити функції з інших служб або подій. Програма може робити прямі запити службі або дії

чи запросити в Android зареєстровані служби і програми. Екземпляр класу Intent являється структурою даних, що містить описання операції, яка повинна виконатися та зазвичай використовується для запуску Activity або сервісу. У випадку з приймачами, цей об’єкт містить опис події, яка відбулася або була оголошена. Система Android у відповідь на намір знаходить відповідний компонент: Activity, Service або Broadcast Receiver та запускає його якщо це необхідно. Повідомлення-намір, яке відправлене певному компоненту, буде отримано тільки цим компонентом. Наприклад, коли одна Activity запускає іншу, то для цього вона створює намір з описанням дії та передає його в метод *startActivity()*, а потім система Android перевіряє всі програми на співпадіння з наміром, знаходить потрібну та запускає відповідну Activity, викликавши метод *onCreate()* і передавши в нього об’єкт-намір Intent (*Рис 1.14*) [12].

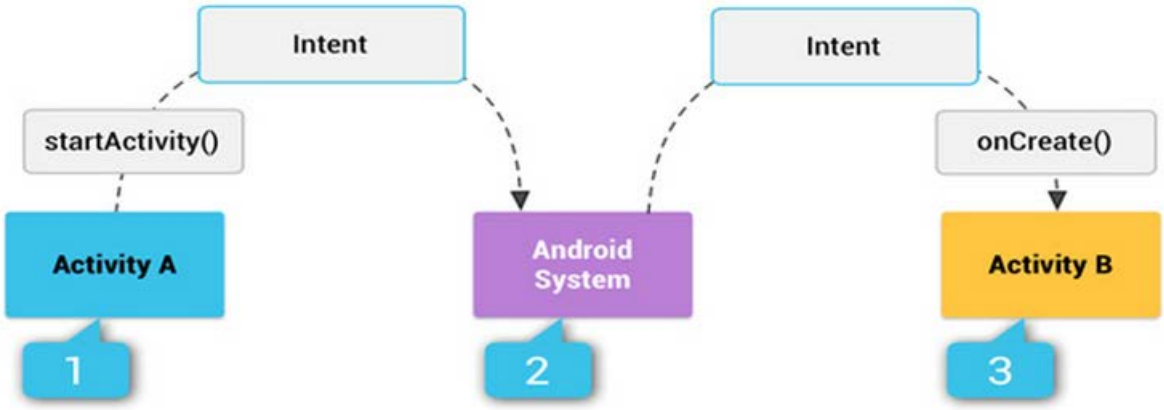

Рис 1.14 – Процес передачі намірів Intent

Таким чином, програма може запросити через Intent, наприклад, програму Контакти для отримання даних або запустити браузер при натисканні посилання в програмі.

• **View** – клас, який являє собою будівельний блок для компонентів інтерфейсу користувача, визначає прямокутну область екрану та відповідає за прорисовку, обробку подій. Є базовим класом для віджетів (кнопки, прапорці, текстові поля та ін.), а також для класу ViewGroup, який являється невидимим контейнером для інших контейнерів та віджетів і визначає властивості розміщення компонентів інтерфейсу користувача. Інтерфейс

Android-програми представляє собою ієрархію цих компонентів (*Рис 1.15*), яку можна створити програмно, але це неправильно. Інтерфейс користувача визначається за допомогою XML, а під час виконання автоматично перетворюється в дерево відповідних об’єктів [8].

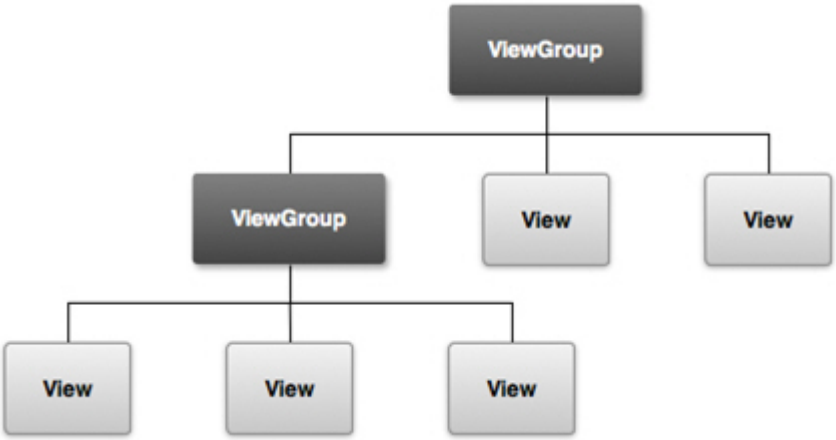

Рис 1.15 – Ієрархія компонентів інтерфейсу користувача Android-програми.

Маніфест Android також є важливою частиною Android-програми, яка представляє собою XML-файл та виконує декілька функцій:

- визначає ім’я Java-пакету програми, яке являє собою унікальний ідентифікатор для програми;

- описує компоненти програми – Activity, Services, Broadcast Receivers та Content Providers. Визначає назви класів, які реалізують кожен із компонентів та оголошує їх можливості (наприклад, які Intent-повідомлення вони можуть опрацьовувати). Ці оголошення дозволяють ОС Android знати, які компоненти при яких умовах можуть бути запущені;

- передбачає процеси, які будуть містити компоненти програми;

- оголошує дозволи, які потрібні для доступу до компонентів програми;

- оголошує дозволи, які програма повинна мати для доступу до захищених частин API та взаємодіяти з іншими програмами;

- оголошує мінімальний рівень API, який потребує програма;

- перераховує бібліотеки, с якми програма повинна бути пов’язана.

Android-програми використовують ресурси, такі як зображення, шари графічного інтерфейсу користувача, оголошення меню (XML-файли) та текстові рядки. Ці ресурси ідентифікуються рядками, які можуть містити,

наприклад, шлях та назву файлу, що містить зображення. В проектуванні Android-програми створюється спеціальний Java-клас з назвою R, який містить static final набори даних. Кожен з таких наборів являється посиланням на окремий ресурс, який використовується в коді програми для зв'язку з ресурсами. Це допомагає виявити помилки в посиланні на ресурс при компіляції, наприклад, коли розробник написав посилання на неіснуючий ресурс або допустив помилку в ньому.

Android-програма використовує декілька різних типів файлів: файли загального призначення, бази даних, кешовані файли та файли Opaque Binary Blob (являють собою зашифровану файлову систему, яка може бути змонтована для програми) [11].

1.2.2 Середовище розробки Android Studio.

Перша Android Studio (*Рис 1.16*) версії 0.1 з'явилася ще в травні 2013 і до червня 2014 вже мала версію 0.8, а в грудні 2014 розробники побачили повноцінну версію 1.0. Пакет інсталяції цього середовища можна завантажити з сайту «*http://developer.android.com/sdk/index.html*», який включає в себе необхідний мінімум. При необхідності можна запустити Android SDK Manager (*Рис 1.17*) та провірити наявність нових версій SDK [12].

Рис 1.16 – логотип Android Studio

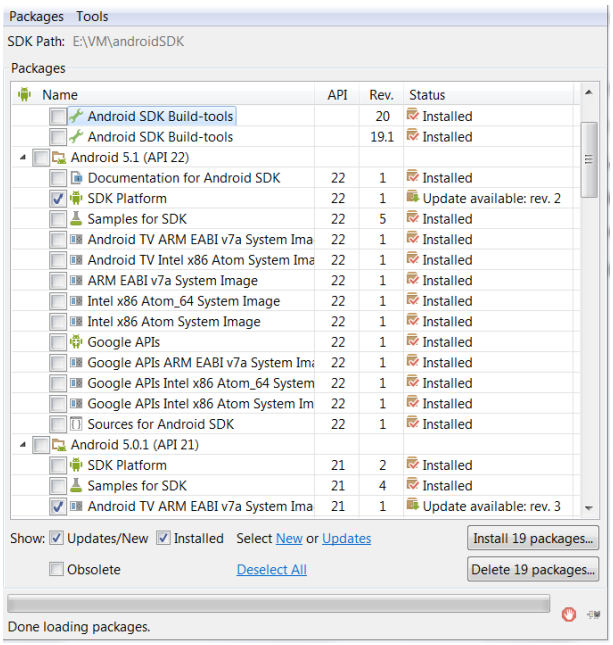

Рис 1.17 – Android SDK Manager

Для відлагодження програм використовується емулятор, який є віртуальною машиною на якій запускається програма. Також можна використовувати і реальний пристрій. Для створення емулятору необхідно перейти в меню «***Tools→Android→AVD Manager***». Відкриється діалогове вікно, де необхідно натиснути на кнопку «***Create Virtual Device***» та в новому вікні відкриється набір можливих емуляторів (*Рис 1.18*).

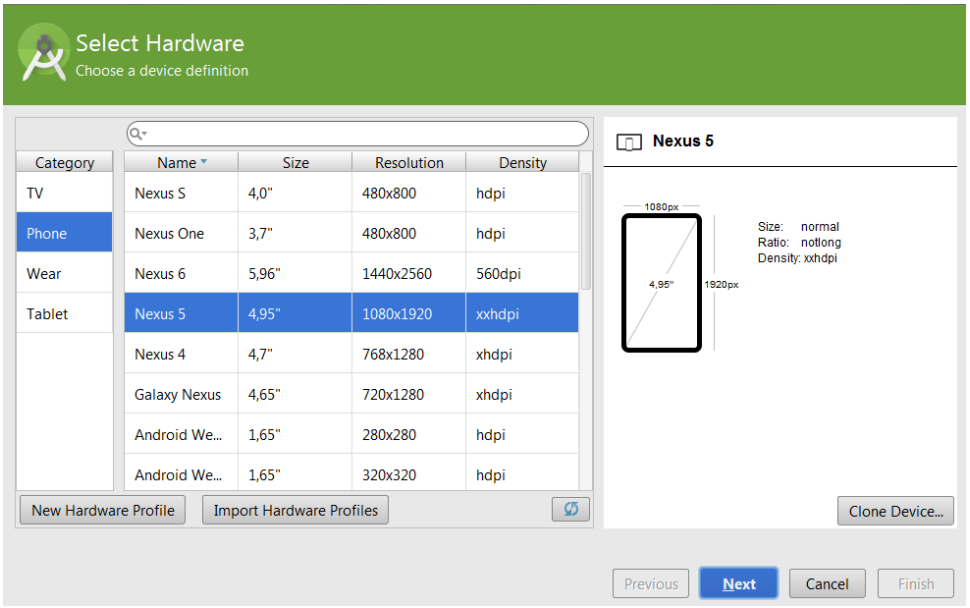

Рис 1.18 – Набір можливих емуляторів.

Вибравши емулятор і натиснувши на «***Next***», відкриється вікно з вибором API (*Рис 1.19*). Додаткові API при необхідності можна завантажити через SDK Manager.

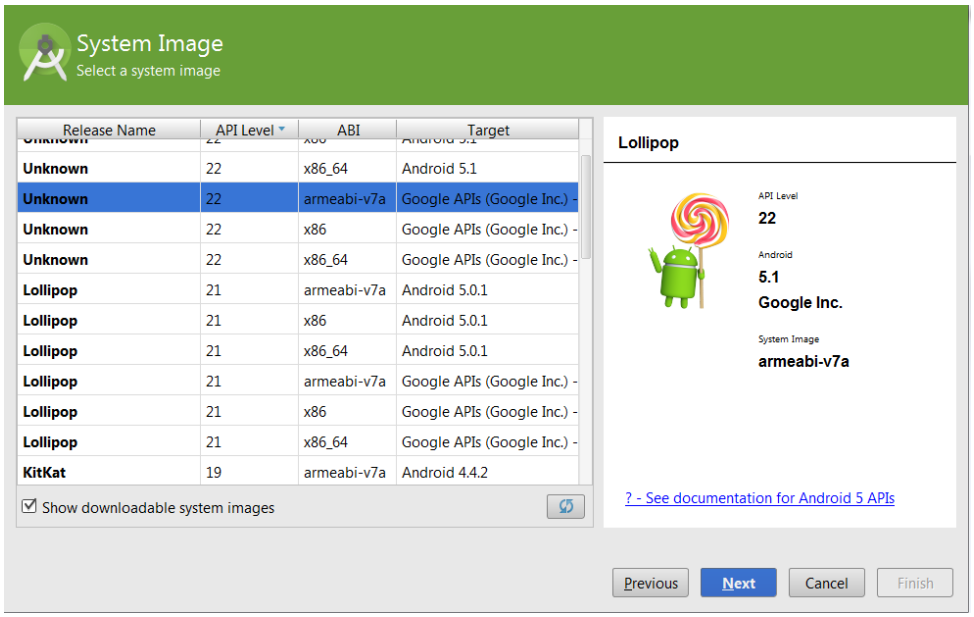

Рис 1.19 – Список доступних API.

Після вибору емулятора та API для нього відкриється вікно, де можна змінити дані та вибрати орієнтацію, з якою пристрій буде запускатися (*Рис 1.20*). Після натискання на копку «***Finish***», новостворений пристрій буде доданий в менеджер емуляторів (*Рис 1.21*).

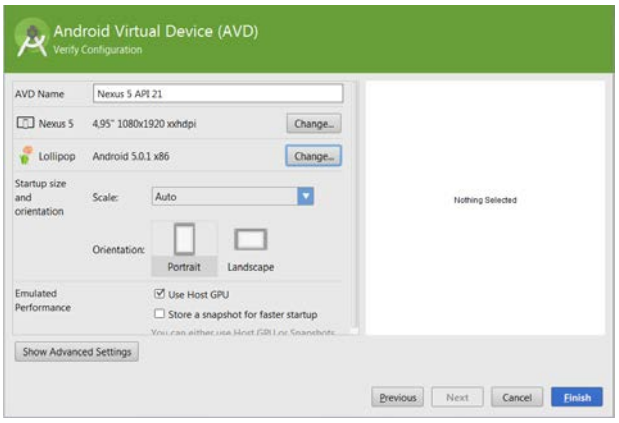

Рис 1.20 – Вікно підтвердження даних.

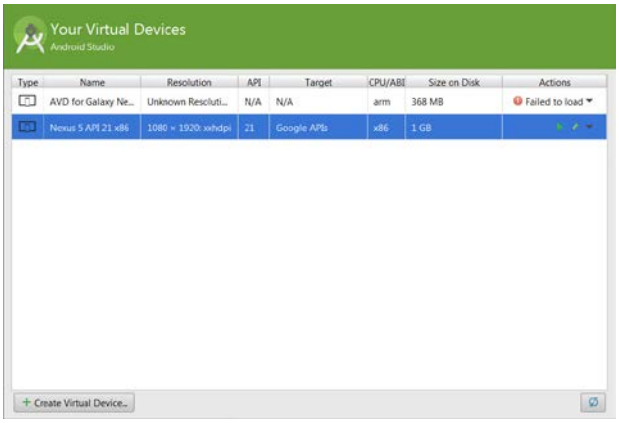

Рис 1.21 – Менеджер емуляторів.

З менеджера емуляторів можна запускати пристрій або змінювати його налаштування. Зазвичай емулятор не запускають окремо, бо коли розробник буде запускати програму, то середовище саме запропонує його запустити. Віртуальні машини зберігаються в папці, яку обрав користувач та шлях до якої повинен містити тільки латинські літери, щоб не викликати проблем [9].

Коли програма повністю розроблена, то її кінцевий варіант бажано перевірити на реальному пристрої. Для цього на ньому потрібно в налаштуваннях увімкнути режим розробника та вибрати там прапорець режиму налагодження через USB. Для деяких пристроїв потрібен окремий драйвер, який потрібно шукати на сайті виробників [10].

1.2.3 Створення проекту на Android Studio та його структура.

В якості мови програмування для Android використовується Java, а для створення інтерфейсу користувача – XML. В середовище вбудована програма «Hello World», яка автоматично створюється разом з проектом. Щоб перевірити, чи всі компоненти середовища встановлені правильно і розробник може створювати програми та відлагоджувати їх, необхідно запустити «Hello World».

Створити новий проект можна натиснувши вкладку «***File→New Project…***», після чого відкриється діалогове вікно майстра (*Рис 1.22*).

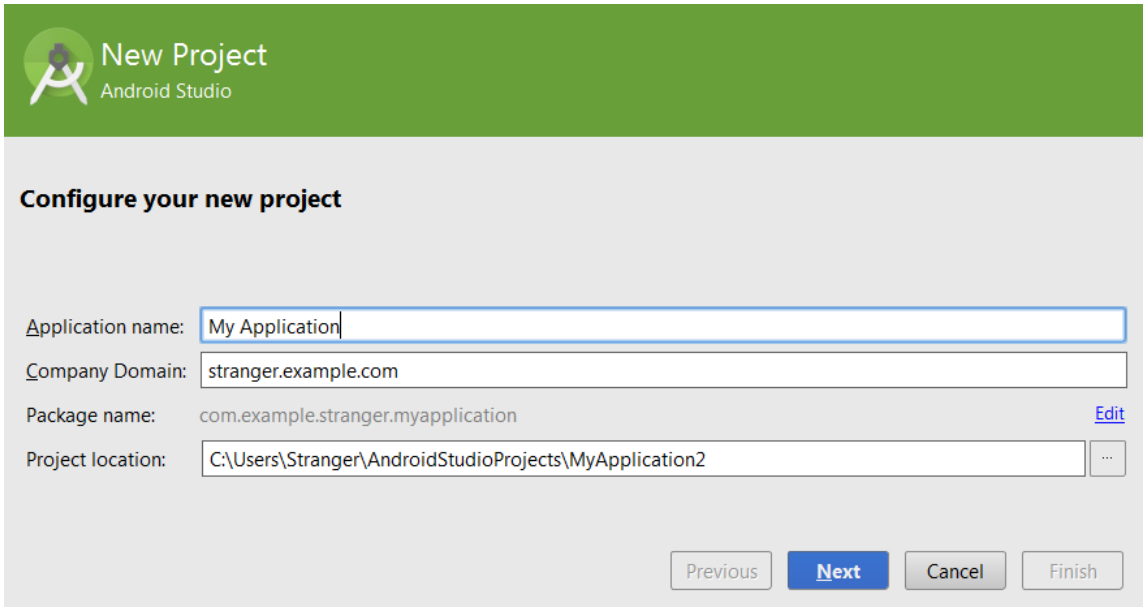

Рис 1.22 – Майстер створення проекту Android

Application Name – ім'я для програми, яке буде відображатися в її заголовку. В початкових налаштуваннях завжди використовуються латинські назви, а локалізовані готовляться пізніше.

Company Domain – поле, що служить для вказування сайту. За замовчуванням там з'являється ім'я користувача комп'ютера. Якщо є сайт, то можна ввести його адресу. Введена назва запам'ятовується та буде автоматично додаватися при створенні нових проектів.

Package name – поле, що формулює спеціальний Java-пакет на основі назви з попередніх полів. В Java використовується перевернутий варіант для назви пакетів, тому спочатку йде «**com**», а потім назва сайту. Пакет служить для унікальної ідентифікації програми, якщо її будуть розповсюджувати. Якщо багато розробників напишуть програму з однаковим ім'ям, то буде незрозуміло, де програма, написана конкретним розробником. А програму з назвою пакету «*<домен><назва_сайту><ім'я_програми>*» буде легше знайти. Google в своїй документації використовує пакет «**com.example**» з метою демонстрації. Якщо розробник буде копіювати приклади з документації та намагатися викласти в магазин Google Play, то в нього не

вийде це зробити, бо ця назва зарезервована та заборонена до використання в магазині програм. Кнопка «**Edit**» дозволяє відредагувати підготовлений варіант, наприклад, на випадок, коли розробник розробляє програму на замовлення і йому необхідно використовувати ім'я пакету, яке затверджено замовником.

Project Location – поле, яке дозволяє вибрати місце на жорсткому диску для проекту, що створюється.

Натиснувши на кнопку «**Next**» відкривається слідуюче вікно, де можна вибрати типи пристроїв, під які буде розроблятися програма (*Рис 1.23*). Окрім смартфонів та планшетів, можна писати програми і для Android TV, Android Wear та Glass, для якого потрібно додатково встановити SDK. Також в цьому ж вікні потрібно вибрати мінімальну версію системи, на якій буде працювати програма. Раніше, до літа 2014 рекомендувалося ставити підтримку Android 2.2, з літа і до кінця 2014 було Android 4.0, а з початку 2015 вже рекомендують ставити підтримку Android 4.1.

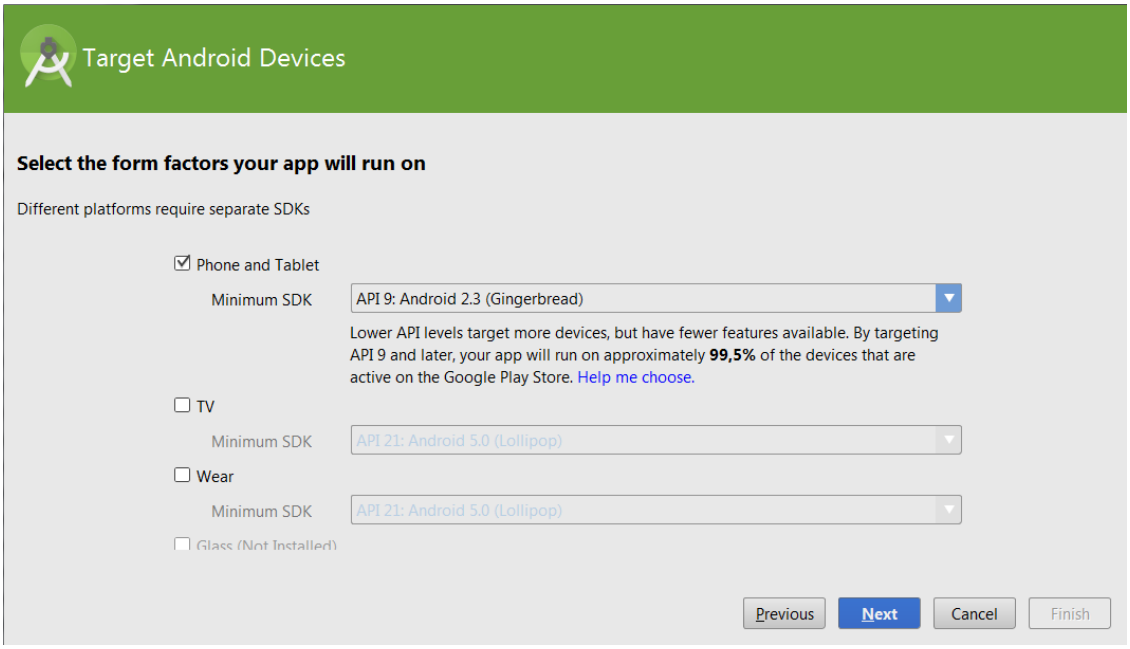

Рис 1.23 – Вікно вибору типу пристроїв, для яких програма буде розроблятися.

Можна натиснути посилання «Help me to choose», після чого відкриється вікно з графіком версій, де будуть зображені відсотки пристроїв, на яких буде працювати програма після вибору тієї чи іншої мінімальної версії системи [13].

Натиснувши «***Next***», відкриється вікно, де потрібно вибрати зовнішній вигляд програми (*Рис 1.24*).

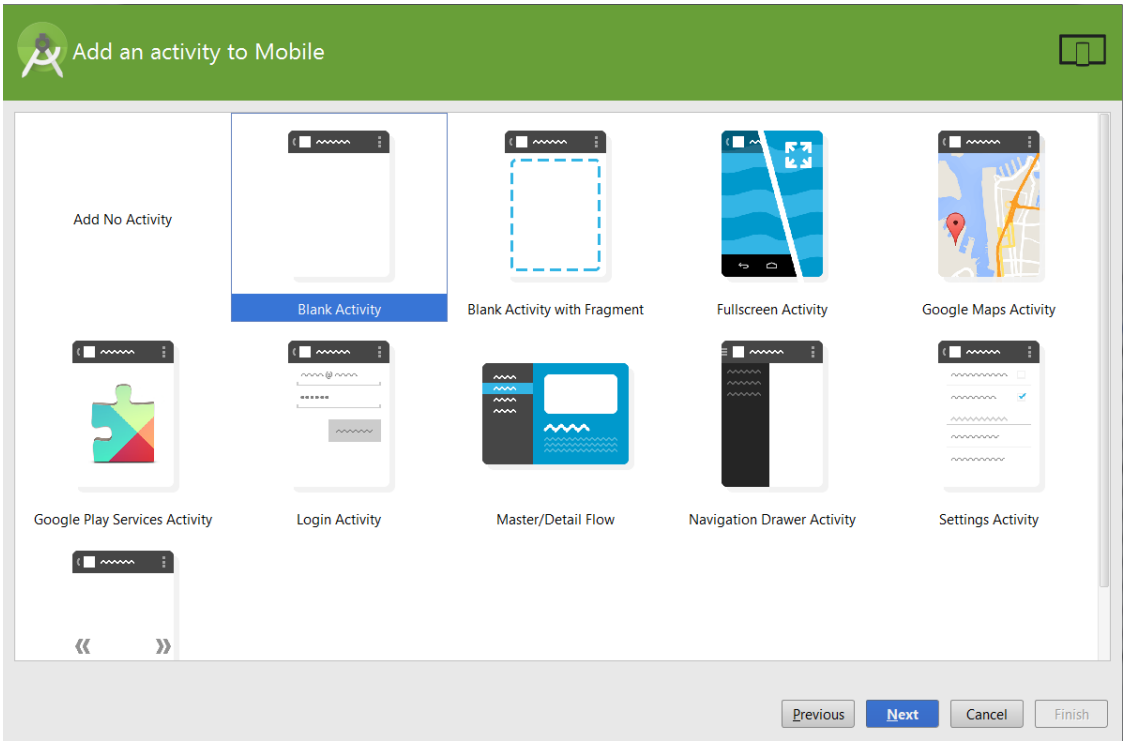

Рис 1.24 – Вікно вибору зовнішнього вигляду програми.

Шаблони, що пропонуються програмою, дозволяють зекономити час для написання стандартного коду для типових ситуацій. Досвідчений розробник може написати вручну будь-який із запропонованих варіантів. Для цього він може вибрати **Add No Activity**, де не буде ніяких шаблонів. Список шаблонів постійно поповнюється новими, а декілька років тому існував тільки один. **Blank Activity** призначений для звичайних телефонів. На зображенні шаблону можна побачити приблизний вигляд програми з його використанням. **Master/Detail Flow** призначений для планшетів з реалізацією двохпанельного режиму. **Fullscreen Activity** можна використати для написання ігор, коли потрібен простір без лишніх деталей. Інші шаблони потрібні для створення програм, які працюють з картами або сервісами Google Play [9].

Після натискання «**Next**» відкриється вікно із заголовками і в ньому вже можна завершити створення проекту, натиснувши кнопку «**Finish**».

Середовище формує проект та створює необхідну структуру з різних файлів та папок (*Рис 1.25*).

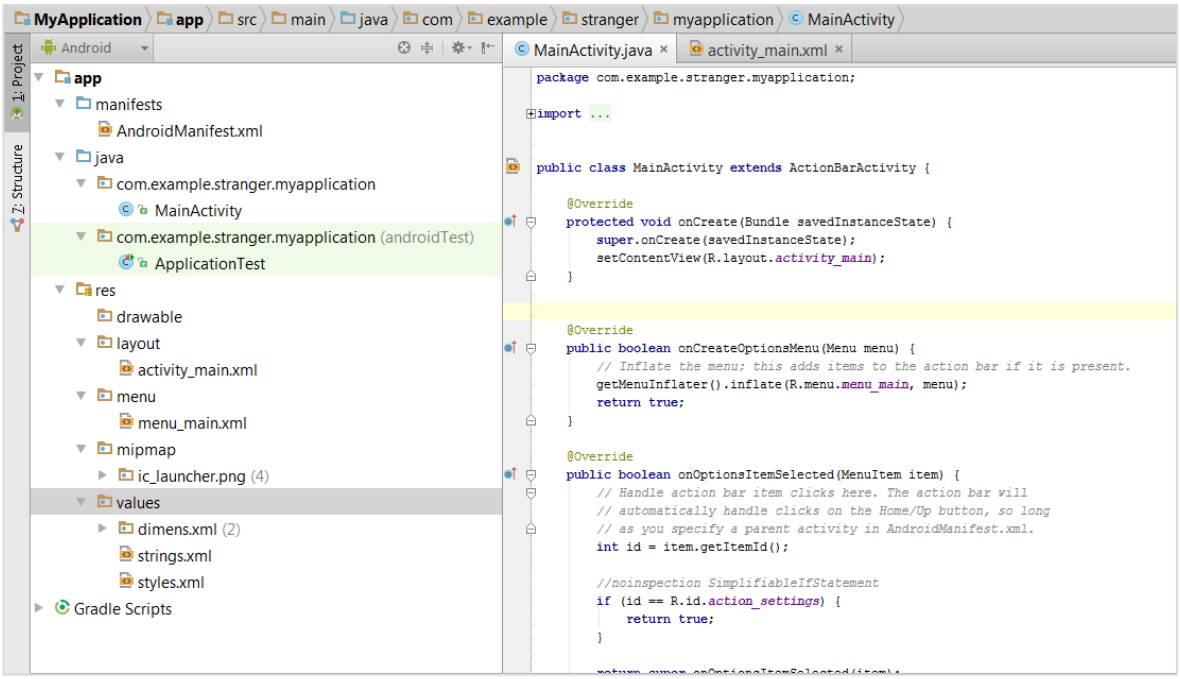

Рис 1.25 – Створений проект та його файлова структура.

Насамперед створюються дві основні папки: **app**, яка містить всі необхідні файли програми та **Grandle Scripts**, що слугує для різних налаштувань, управління проектом і т.д. В папці **app** містяться ще три каталоги:

• **manifest**, що має в собі один файл *AndroidManifest.xml*, в якому повинні бути оголошені всі Activity, Services, Broadcast Receivers та Content Providers програми. Також він містить необхідні програмі дозволи, такі як, наприклад, доступ до мережі, якщо це потрібно програмі.

• **java**, що містить дві підпапки, одна з яких робоча і містить файли класів, а інша створена для тестів.

• **res**, що містить файли ресурсів, які розподілені по окремим підпапкам, таких як: **dwarable** (в цих каталогах містяться графічні ресурси, призначені для різних розмірів екрану), **layout** (тут містяться xml-файли, які описують зовнішній вигляд форм, та їх різних елементів; при створенні проекту там вже є створений файл *activity_main.xml*, що відповідає за вигляд головного Activity програми), **menu** (в цій папці знаходяться ресурси для

меню; тут вже є створений файл *menu_main.xml* і він відповідає за меню головного Activity програми), **mipmap** (ця папка зберігає різні іконки програми і по своїй структурі співпадає з dwarable, в якій раніше все це містилося і тепер вона використовується для інших графічних ресурсів) та **values** (в цьому каталозі містяться будь-які рядкові ресурси та ресурси тем, кольорів, стилів і т.п., що використовуються в проекті).

Android підтримує спосіб, що заснований на XML-розмітці. Файли розмітки являються деревом XML-елементів, де кожен вузол є іменем класу View та знаходяться в папці «*res/layout*». При розробці програми розробник додає різні компоненти, наприклад, кнопки, текстові поля, прапорці і т.д, які потім будуть відображуватися на екрані. При цьому він використовує структуру і синтаксис XML Дизайн програми можна робити візуально, перетягуючи об’єкти за допомогою миші (вкладка Design в середовищі Android Studio) (*Рис 1.26*), та прописувати вручну (вкладка Text) (*Рис 1.27*). До прикладу можна розглянути файл *activity_main.xml*, що відповідає за дизайн головного Activity [13].

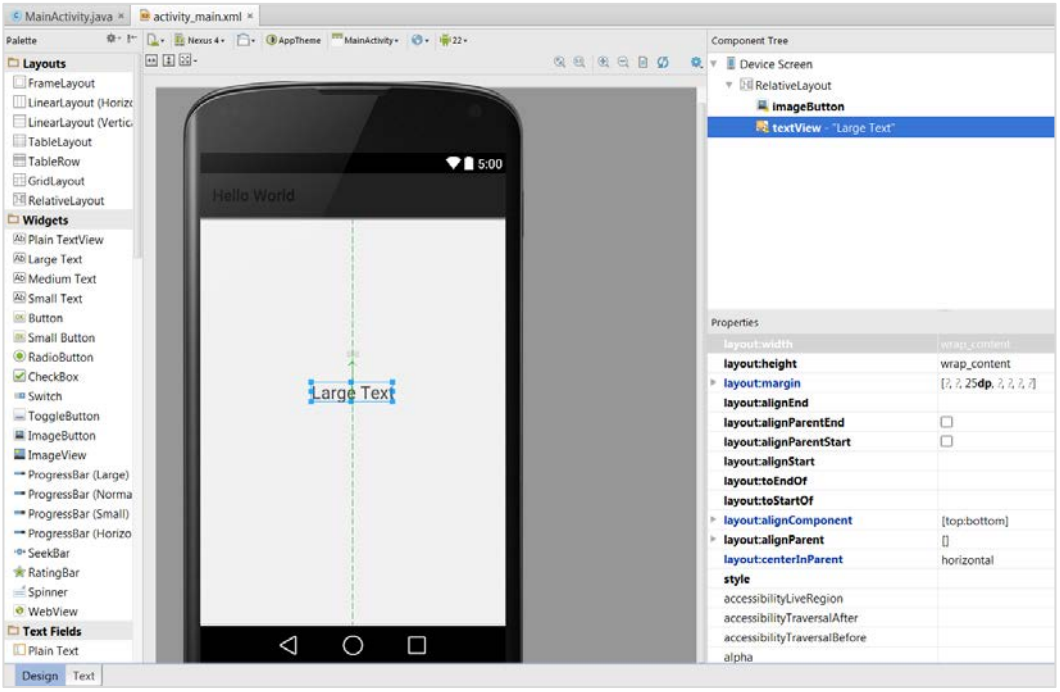

Рис 1.26 – Графічне представлення файлу розмітки

При графічному режимі зліва від основної частини редактора можна побачити панель інструментів, де згруповані різні елементи по групах. Їх

можна переміщати на форму за допомогою миші. З правої сторони знаходяться вкладки ***Component Tree***, де представлене дерево компонентів, що є на формі, та ***Properties***, де відображаються різні властивості вибраного елемента. На Рис 1.26 видно, що графічним способом було додано компоненти *imageButton* та *textView***.**

При текстовому перегляді також з правої частини можна побачити візуальне представлення екрану, тобто як він буде виглядати при зміні тієї чи іншої властивості або додаванням нового компоненту.

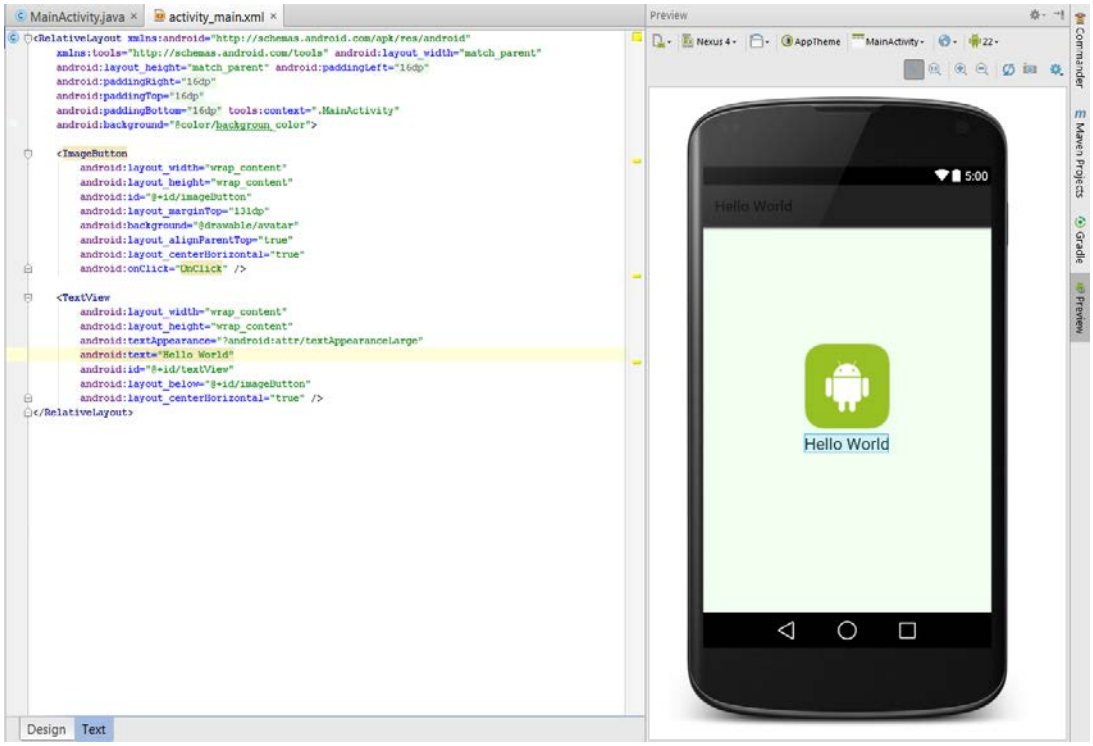

Рис 1.27 – Текстове представлення файлу розмітки.

На Рис 1.27 можна побачити, що вручну було змінено поле *text* компоненту *textView*, додано зображення в *imageButton* шляхом прописування в полі *background* його місцезнаходження (папка *dwarable*). Також Був змінений фон головного екрану в полі *background*.

Створивши проект, необхідно запустити його на емуляторі мобільного пристрою, щоб побачити правильність роботи програми. Для цього потрібно натиснути на зелену кнопку «***Run***» або натиснути комбінацію клавіш «***Shift+F10***». Відкриється вікно з вибором віртуального пристрою (*Рис 1.28*), вибравши який середовище його завантажить (*Рис 1.29*).

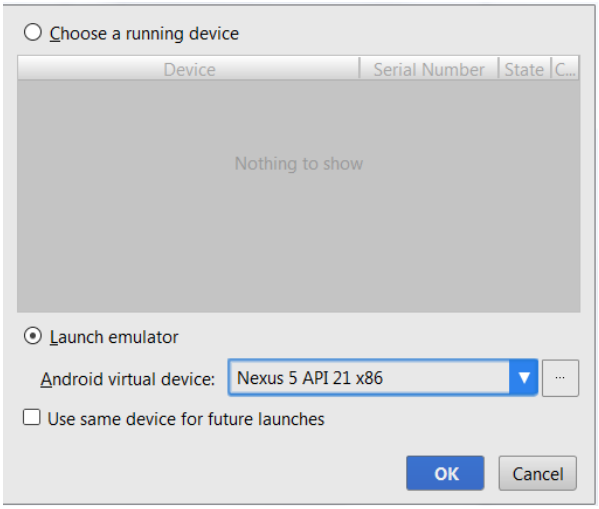

Рис 1.28 – Вікно вибору емуляторів

Рис 1.29 – Завантажений емулятор.

Середовище завантажує емулятор пристрою та встановлює туди програму проекту, яку автоматично запускає. Також її можна запустити вручну, натиснувши на її іконку в меню системи (*Рис 1.29*). В даному випадку встановилася програма «Hello World», запустивши яку можна перевірити її функціональність (*Рис 1.30*).

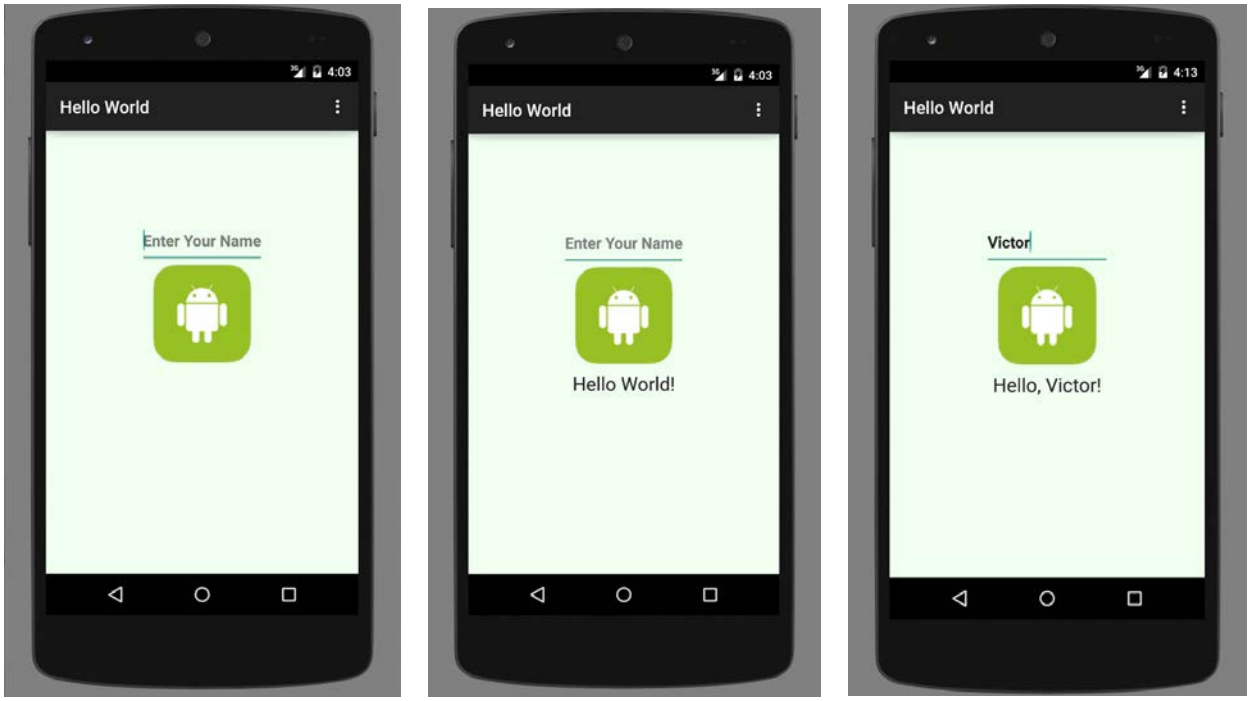

Рис 1.30 – Програма «Hello, World», запущена на емуляторі.

В даній програмі є текстове поле для введення з підказкою «Enter Your Name» та зображення-кнопка, натиснувши на яку в текстове поле нижче виводиться значення «Hello, <*your_name*>!». Якщо не було нічого введено, то в текстове поле просто виведеться «Hello World!» [10].

РОЗДІЛ 2. РОЗРОБКА ПРОЕКТУ «ЕЛЕКТРОННИЙ ПОСІБНИК»

2.1 Інтерфейс.

Інтерфейсна частина в Android може будуватися на звичайних xml-файлах, або за допомогою java-коду.

Основний шар, що створивсь автоматично знаходиться в папці «*res/layout/activity_main.xml*»

Після подвійного клацання лівою кнопкою миші відкриється візуальний редактор шару. Можна обійтись без нього, перейшовши на вкладку «*Text*» в нижній частині вікна, де можна вручну відредагувати XML.

Спочатку потрібно видалити автоматично створений TextView і замість нього додати ImageView для відображення Splash-зображення:

```
<ImageView
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:id="@+id/splashscreen"
    android:layout_gravity="center"
    android:src="@drawable/splash"
    />
```

• layout_width і layout_height – ширина і висота елемента відповідно, fill_parrent показує, що елемент буде розтягуватися на всю площу батьківського елемента.

• id – визначає унікальний ідентифікатор елемента.

• layout_gravity – розміщує Splash-зображення ро центру.

• src – підключає зображення, яке знаходиться в папці графічних ресурсів dwarable.

Далі потрібно додати ListView для відображення списку версій, що також розтягується на всю площу:

```
<ListView
    android:id="@+id/listView"
    android:layout_height="fill_parent"
```

```
        android:layout_width="fill_parent"
        android:background="#fff0f0f0"
        />
```

По завершенню *activity_main.xml* повинен виглядати так:

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/splashscreen"
        android:layout_gravity="center"
        android:src="@drawable/splash"
         />

    <ListView
        android:id="@+id/listView"
        android:layout_height="fill_parent"
        android:layout_width="fill_parent"
        android:background="#fff0f0f0"
         />
    </LinearLayout>
```

Далі потрібно створити шар для другого Acivity. Для цього створюється новий файл в папці «*layout*». Нехай буде під назвою *activity_view.xml*.

Сюди потрібно додати тільки WebView:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
     android:orientation="vertical"
     android:layout_width="fill_parent"
     android:layout_height="fill_parent">

     <WebView
          android:id="@+id/webView"
          android:layout_width="fill_parent"
          android:layout_height="fill_parent"
          android:background="#fff0f0f0"
          />
     </LinearLayout>
```

2.2 Головне Activity.

В головному Activity буде показуватися Splash-зображення і список тем. Файл буде називатися *WebsterActivity.java*.

```
package com.example.stranger.posibnuk;

import android.content.Intent;
import android.os.Handler;
import android.os.Message;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ImageView;
```

```
import android.widget.ListView;
public class MainActivity extends ActionBarActivity
    {
    private ListView lv1;
        //Даємо назву для listview - lv1.
    private ImageView splash;
    private static final int STOPSPLASH = 0;
    private static final long SPLASHTIME = 4000;
        //Час показу Splash-зображення.
    private String lv_arr[]={
            "Вступ",
            "1. Android 1.0 Apple Pie.",
            "2. Android 1.2 Banana Bread.",
            "3. Android 1.5 Cupcake.",
            "4. Android 1.6 Donut.",
            "5. Android 2.0, 2.1 Enclair",
            "6. Android 2.2 Froyo.",
            "7. Android 2.3 Gingerbread.",
            "8. Android 3.0 Honeycomb.",
            "9. Android 4.0 Ice Cream Sandwich.",
            "10. Android 4.1, 4.2, 4.3 Jelly Bean.",
            "11. Android 4.4 KitKat.",
            "12. Android 5.0 Lollipop"
        };
        //Створюємо масив з тем.
    private Handler splashHandler = new Handler()
        {
        //створюємо новий хендлер.
        @Override
```

```
        public void handleMessage(Message msg)
             {
             switch (msg.what)
                  {
                  case STOPSPLASH:
                       splash.setVisibility(View.GONE);
        //забираємо Splash зображення - міняємо видимість
                  break;
                  }
             super.handleMessage(msg);
             }
        };

    @Override
    protected void onCreate(Bundle savedInstanceState)
         {
         super.onCreate(savedInstanceState);
         setContentView(R.layout.activity_main);
         //ставимо шар головного екрану.
         splash = (ImageView)
                             findViewById(R.id.splashscreen);
//отримуємо ідентифікатор ImageView із Splash-зображенням.
         Message msg = new Message();
         msg.what = STOPSPLASH;
         splashHandler.sendMessageDelayed(msg, SPLASHTIME);
         lv1 = (ListView)findViewById(R.id.listView);
         //отримуємо ідентифікатор ListView.
         lv1.setAdapter(new ArrayAdapter<String>(this,
             android.R.layout.simple_list_item_1 , lv_arr));
```

```
            //встановлюємо масив в ListView.
            lv1.setTextFilterEnabled(true);
            //Обробляємо натискання по пункту:
            lv1.setOnItemClickListener(
                                    new OnItemClickListener()
                {
                public void onItemClick(AdapterView<?> a,
                                View v, int position, long id)
                    {
            //Позиція елемента, на який натиснули:
                    String itemname = new Integer(position).
                                    toString();
            //Створюємо intent:
                    Intent intent = new Intent();
                    intent.setClass(MainActivity.this,
                                        ViewActivity.class);
                    Bundle b = new Bundle();
                    b.putString("defStrID", itemname);
//defStrID - унікальний рядок, відправимо itemname в друге
Activity:
                    intent.putExtras(b);
            //запускаємо Intent
                    startActivity(intent);
                    }
                });
            }
        }
```

2.3 Друге Avtivity.

Друге Activity містить тільки WebView, яке буде відображувати html текст. Текстові файли в іменами *n0.txt, n1.txt,…,n10.txt* переміщуються в папку «*res*/*raw*» (*Рис 2.1*).

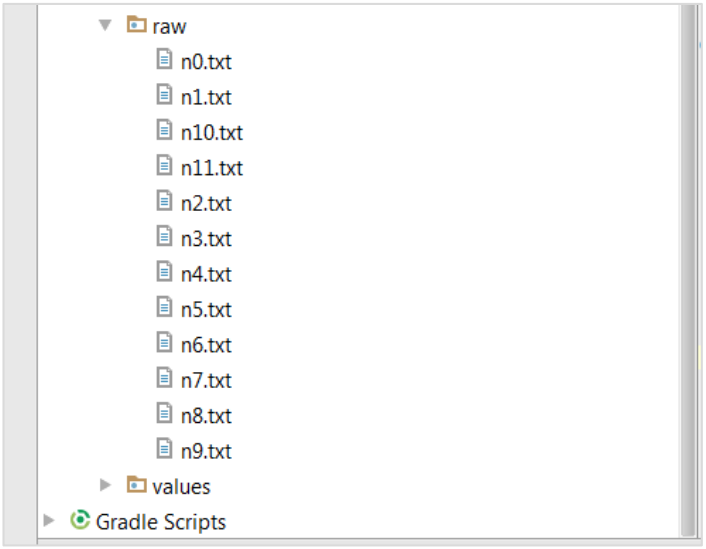

Рис 2.1 – Структура папки raw.

Створюється файл з назвою ViewActivity.java:

```
package com.example.stranger.posibnuk;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.view.Window;
import android.view.WindowManager;
import android.webkit.WebView;
public class ViewActivity extends Activity
    {
    @Override
```

```
public void onCreate(Bundle savedInstanceState)
    {
    super.onCreate(savedInstanceState);
    requestWindowFeature(Window.FEATURE_NO_TITLE);
        //приховуємо заголовок.
    setContentView(R.layout.activity_view);
        //приховуємо статусбар.
    getWindow().setFlags(
        WindowManager.LayoutParams.FLAG_FULLSCREEN,
          WindowManager.LayoutParams.FLAG_FULLSCREEN);
    Bundle bundle = getIntent().getExtras();
    String itemname = "n" +
                        bundle.getString("defStrID");
        //отримуємо рядок і формуємо ім'я ресурса.
    Context context = getBaseContext();
    //отримуємо контекст.
    //читаємо текстовий файл із ресурсів по імені:
    String text = readRawTextFile(context,
        getResources().getIdentifier(itemname,
             "raw", "com.example.stranger.posibnuk"));
    WebView myWebView = (WebView)
                             findViewById(R.id.webView);
    String summary = "<!Doctype html><html><body>" +
                            text + "</body></html>";
    myWebView.loadDataWithBaseURL(null,summary,
                            "text/html" ,"utf-8",null);
    //завантажуємо текст в webview
    }
```

```
    public static String readRawTextFile(Context ctx,
                                                    int resId)
//читаємо текст із raw – аргументи, контекст та
                                        ідентифікатор ресурсу.
    {
        InputStream inputStream = ctx.getResources().
                                        openRawResource(resId);
        InputStreamReader inputreader =
                        new InputStreamReader(inputStream);
        BufferedReader buffreader =
                        new BufferedReader(inputreader);
        String line;
        StringBuilder text = new StringBuilder();
        try
        {
            while (( line = buffreader.readLine()) !=
        null)    {
                    text.append(line);
                    text.append('\n');
                    }
        }
        catch (IOException e)
            {
            return null;
            }
        return text.toString();
    }
}
```

2.4 Маніфест.

Для того, щоб програма працювала, потрібно налаштувати маніфест. Необхідно відкрити *AndroidManifest.xml* для написання коду в нього.

Крім описання головного Activity:

```
<activity
     android:name=".MainActivity"
     android:label="Історія Android">
          <intent-filter>
               <action android:name="android.
                                   intent.action.MAIN" />
               <category android:name="android.
                                   intent.category.LAUNCHER" />
               </intent-filter>
     </activity>
```

Потрібно додати друге Activity:

```
<activity
     android:name=".ViewActivity"
     android:label="Перегляд">
          <intent-filter>
               <action android:name=".ViewActivity" />
               <category
                    android:name="android.
                    intent.category.DEFAULT"></category>
               </intent-filter>
     </activity>
```

Повністю маніфест буде мати вигляд:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:
     android="http://schemas.android.com/apk/res/android"
     package="com.example.stranger.posibnuk" >
```

```
    <application
        android:allowBackup="true"
        android:icon="@mipmap/icon_lau"
        android:label="Електронний посібник"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="Історія Android" >
            <intent-filter>
                <action android:
                name="android.intent.action.MAIN" />
                <category android:
                name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".ViewActivity"
            android:label="Перегляд">
            <intent-filter>
                <action android:
                name=".ViewActivity" />
                <category android:
                name="android.intent.category.DEFAULT">
                </category>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

2.5 Ресурси.

Тепер необхідно імпортувати ресурси для програми – Splash-зображення, яке треба помістити в папку «*drawable*», та іконку, яку слід помістити в папку «*mipmap*». Іконки є різних розмірів, кожен з яких призначений для конкретного розміру екрану.

Splash-зображення та іконку можна створити в графічному редакторі або завантажити з мережі (*Рис 2.2*).

Рис 2.2 – Іконка та Splash-зображення проекту

2.6 Запуск програми на емуляторі.

Після завершення всіх кроків можна запустити програму на віртуальному Android-пристрої (*Рис 2.3*), (*Рис 2.4*):

Рис 2.3 – Проінстальована програма на пристрої, Splash-зображення, заголовок програми

Рис 2.4 –Список зі вступом і версіями та відритий вступ.

Після запуску програми з’являється Splash-зображення, яка потім зникає та залишається список версій. Після натискання на елемент списку буде завантажуватися друга Activity з WebView, в якій буде завантажуватися контент із текстового файлу. В ній будуть приховані заголовок програми на рядок стану.

ВИСНОВКИ

В наш час складно уявити собі людину без мобільного телефону, планшетного комп'ютера, смартфона або будь-якого іншого портативного мультимедійного пристрою. Ми звикли до того, що завжди під рукою не тільки засіб зв'язку, але і багато різноманітних функцій, таких як: калькулятор, органайзер, конвертер, календар, годинник. Смартфони становляться новою мобільною ігровою платформою, змагаючись з класичними карманними ігровими системами на прикладі Nintendo DS або Playstation Portable.

У влаштуванні смартфона все достатньо просто. Головним нюансом є те, що він складається з декількох ріхних блоків – пам'яті, процесора, який займається обчисленням, сховища даних, радіомодуля, який в свою чергу складається з приймача і передавача та відповідає за зв'язок. Найцікавіше тут – операційна система встановлена на вмонтовану пам'ять. Від неї та її версії залежать всі основні можливості пристрою. Смартфони, як і персональні комп'ютери, існують в абсолютно різних комплектаціях та під управлінням різних операційних систем.

Android являється однією із найновіших розробок серед операційних систем. Вона призначена для широкого кола мобільних пристроїв. Операційна система Android встановлюється найчастіше на комунікатори, а також на планшетні ПК, смартбути, нетбуки.

Розробник цієї системи – компанія Android Inc, яка пізніше була прекуплена компанією Google. В даний час розробками і розвитком систем на базі Android займається компанія Open Handset Alliance. Вона включає в себе не тільки Google, але і Motorola, HTC, Intel, Samsung та багато інших гігантів в галузі розробки техніки.

Операційна система розроблена на базі Linux, але містить не всі розробки. Це зв'язано з використанням віртуальної машини Dalvik. Саме на ній відбувається робота всього програмного обладнання.

У дипломній роботі розглянуті версії мобільної платформи Android починаючи від 1.0 Apple Pie та закінчуючи 5.0 Lollipop.

Досліджена архітектура мобільної програми під Android, розглянуто її основні компоненти, що є блоками для побудови програми.

Також у даній роботі представлене середовище розробки Android-програм, підготовка до їх написання, створення віртуального пристрою операційної системи, сворення проекту, його налаштування і запуск на емуляторі мобільного пристрою з ОС Android.

За допомогою середовища розробки Android-програм був розроблений проект «Електронний посібник», який дозволяє встановити його на мобільний пристрій з ОС Android та перегрядати з нього інформацію. Цей проект був розроблений мовою Java в середовищі Android Studio.

Java являється об’єктно-орієнтованою мовою програмування. Синтаксис програмного коду для ПК та мобільних пристроїв відрізняється, оскільки в Android SDK використовується багато власних бібліотек, в першу чергу тих, які забезпечують взаємодію користувача з пристроєм жестами.

Розробники не стоять на місці. Операційна система Android постійно удосконалюється і впроваджується у всі найновіші види техніки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Google Android [Електронне джерело]. – Режим доступу: URL http://www.androidtalk.ru/google-android/. – Назва з екрану.

2. История версий Android [Електронне джерело]. – Режим доступу: URL http://www.androidpit.ru/Android-G1-Ice-Cream-Sandwich/ – Назва з екрану.

3. Программы и приложения для Android [Електронне джерело]. – Режим доступу: URL http://android-phones.ru/. – Назва з екрану.

4. Мобильная платформа Android [Електронне джерело]. – Режим доступу: URL http://itc.ua/articles/mobilnaya-platforma-android-pyat-let-istorii/. – Назва з екрану.

5. Рік Роджерс, Джон Ломбардо Android. Разработка приложений – М.: ЕКОМ Паблішерс, 2010 – 400 с.

6. Голощапов А. Л. Google Android: программирование для мобильных устройств – СПб.: БХВ-Петербург, 2011 – 448 с.

7. Архитектура Android-приложений [Електронне джерело]. – Режим доступу: URL http://habrahabr.ru/post/141201/. – Назва з екрану.

8. Виды приложений и их структура [Електронне джерело]. – Режим доступу: URL http://www.intuit.ru/studies/courses/12643/1191/lecture/21983?page=2. – Назва з екрану.

9. Освой Android играючи [Електронне джерело]. – Режим доступу: URL http://developer.alexanderklimov.ru/android/android1.php. – Назва з екрану.

10. Android Emulator [Електронне джерело]. – Режим доступу: URL http://developer.android.com/tools/help/emulator.html. – Назва з екрану.

11. Все про Android [Електронне джерело]. – Режим доступу: URL http://androiddocs.ru/ . – Назва з екрану.

12. Android: Теория [Електронне джерело]. – Режим доступу: URL http://developer.alexanderklimov.ru/android/theory/. – Назва з екрану.

13. Учебник по Android [Електронне джерело]. – Режим доступу: URL http://startandroid.ru/ru/. – Назва з екрану.