СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Навчально-науковий інститут (факультет) інформаційних технологій та електроніки
Кафедра програмування та математики

ПОЯСНЮВАЛЬНА ЗАПИСКА
до кваліфікаційної випускної роботи

освітній ступінь бакалавр
спеціальність 121 „Інженерія програмного забезпечення”
(шифр і назва спеціальності)
спеціалізація „Інженерія програмного забезпечення”

на тему „Створення додатку для перегляду текстових файлів в мобільних пристроях”

Виконав: студент групи ІПЗ-16д _____ А.В. Лозовий
(підпис) (ініціали і прізвище)

Керівник _____ Д.М. Марченко
(підпис) (ініціали і прізвище)

Завідувач кафедри _____ В.О. Лифар
(підпис) (ініціали і прізвище)

Рецензент_____

Сєвєродонецьк – 2020

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Навчально-науковий інститут (факультет) інформаційних технологій та електроніки
Кафедра програмування та математики

Освітній ступінь бакалавр
спеціальність 121 „Інженерія програмного забезпечення”
(шифр і назва спеціальності)
спеціалізація „Інженерія програмного забезпечення”
(назва спеціалізації)

ЗАТВЕРДЖУЮ
Завідувач кафедри ПМ,
д.т.н., доцент
_____ Лифар В.О.
“____” _____2020 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ ВИПУСКНУ РОБОТУ СТУДЕНТУ

Лозовий Артем Вікторович
(прізвище, ім’я, по батькові)

1.Тема роботи Створення додатку для перегляду текстових файлів в мобільних пристроях.

Керівник роботи професор, д.т.н.Марченко Дмитро Миколайович,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджений наказом університету від “___”_____20__року №___

2. Строк подання студентом роботи 20 травня 2020 р.

3. Вихідні дані до роботи Об'єктом даної роботи є процес розробки мобільного додатку.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Вступ. Аналітичний огляд, з висвітленням наступних питань: актуальність мобільних пристроїв, інструменти розробки. Основна частина, в якої висвітлити: збір вимог та проектування, етап даталогічного проектування. Висновки. Перелік використаних джерел.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслеників)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання__30 березеня 2020 року._____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання кваліфікаційної випускної роботи	Строк виконання етапів	Примітка
1	Одержання завдання на виконання роботи	30.03.20	
2	Укладання і погодження з керівником плану і етапів виконання роботи	06.04.20	
3	Узагальнення даних літературних джерел, укладання розділу «Аналіз предметної галузі»	13.04.20	
4	Аналіз шляхів виконання завдання. Вибір і погодження керівником оптимального шляху	20.04.20	
5	Укладання та тестування програмного продукту	27.04.20	
6	Укладання, оформлення та погодження пояснювальної записки з керівником	04.05.20	
7	Здача готової пояснювальної записки на кафедру	12.05.20	
8	Укладання доповіді і презентації	30.05.20	

Студент _____ А.В. Лозовий _____
(підпис) (ініціали і прізвище)

Керівник роботи _____ Д.М. Марченко_____
(підпис) (ініціали і прізвище)

ЛИСТ ПОГОДЖЕННЯ І ОЦІНЮВАННЯ

дипломної роботи студента гр. ІПЗ-16д Лозовий А.В.

Науковий керівник

Професор, д.т.н. _____ Марченко Д.М.

Оцінка наукового керівника: _____

Рецензент _____

ПІБ, місто роботи, посада

Оцінка рецензента: _____

Кінцева оцінка за результатами захисту:

_____

Голова ЕК

_____ Лифар В.О.

підпис

РЕФЕРАТ

Робота містить: 40 сторінок основного тексту, 13 сторінок додатків, 18 рисунка, 13 використаних джерел.

Метою випускної кваліфікаційної роботи є виявити більш зручне середовище створення мобільного додатку для перегляду тестових файлів.

Було проаналізовано принцип роботи, перелік наданих можливостей і кращі рішення взаємодії з користувачами.

В результаті виконаної роботи, було розкрито технології розробки мобільного додатку для перегляду тестових файлів та розроблений додаток.

Система реалізована відповідно всім вимогам технічного завдання. Зроблено детальний опис процесу розробки системи, а також, продемонстрована робота готового додатку.

ЗМІСТ

ВСТУП

Актуальність досліджень: Мобільні пристрої, невід'ємна частина життя сучасної людини. Це комунікації між людьми, прослуховування улюбленої музики, перегляд аудіовізуальної інформації, блокнот, і ще безліч функцій. Мобільний смартфон це копія комп'ютера яку завжди можна мати при собі. Які ж причини поширення цієї операційної системи?

По-перше, Android підтримує велику кількість пристроїв різних виробників.

По-друге, Android додатки характеризується високою доступністю засобів створення. Засоби розробки для платформи Android є безкоштовними, в той час як створення додатку, наприклад, під iPhone (від компанії Apple) вимагає вагомих початкових фінансових вкладень. Крім усього перерахованого вище, перевагою ОС Android є наявність безкоштовних бібліотек для роботи зі сторонніми ресурсами (Yandex MapKit, Google Map API, ін.), той час як для Windows Phone Mobile дані бібліотеки не поширені.

Об’єкт досліджень: Розробка мобільного додатку для перегляду тестових файлів.

Предмет досліджень: Технології розробки додатків, який був би легкий у використанні будь-якому користувачу.

Мета дослідження: Виявити більш зручне середовище створення мобільного додатку для перегляду тестових файлів.

Завдання дослідження:

а) скласти інформаційну модель;

б) розкрити технології розробки мобільного додатку для перегляду тестових файлів.

в) створити додаток.

Методологічна та теоретична основа дослідження: методи розробки мобільного додатку для перегляду тестових файлів.

Методи дослідження: технології побудови розробки мобільного додатку для перегляду тестових файлів.

Практичне значення отриманих результатів. Одержаний результат буде використовуватися звичайними користувачами у повсякденному житті.

РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Актуальність мобільних пристроїв

Варто говорити про те, скільки часу звичайна людина проводить зі своїм смартфоном? Близько 8 років за все життя. Практично кожен власник смартфона заходить в інтернет відразу після пробудження вранці і безпосередньо перед відходом до сну. Більшість контенту надходить саме з екранів смартфонів або планшетів. І це не дивно, адже смартфон це маленький пристрій, який завжди під рукою і дає можливість досить легко, натиснувши пару раз на екран отримати доступ до потрібної інформації. Смартфони настільки сильно вкоренилися в нашому житті, що ми не розлучаємося з ними надовго. Підбивши підсумок вищесказаного можна сказати, що смартфони це найпопулярніші цифрові комунікаційні пристрої. Отже, розробка саме мобільного додатка є вірним рішенням.

1.2 Вибір операційної системи

На даний момент існує різних мобільних багато операційних систем. Але всі вони мають різну популярність. Якщо програма не кроссплатформне, то розробка для свідомо вузького кола користувачів може заздалегідь приректи його на провал. Адже чим більше користувачів випробує більше додаток, тим залишиться зацікавлених в ньому. Тому одним з головних завдань є вибір операційної системи, під яку буде розроблятися додаток. На рис. 1.1 представлена діаграма показує частки які займають різні мобільні операційні системи на ринку.

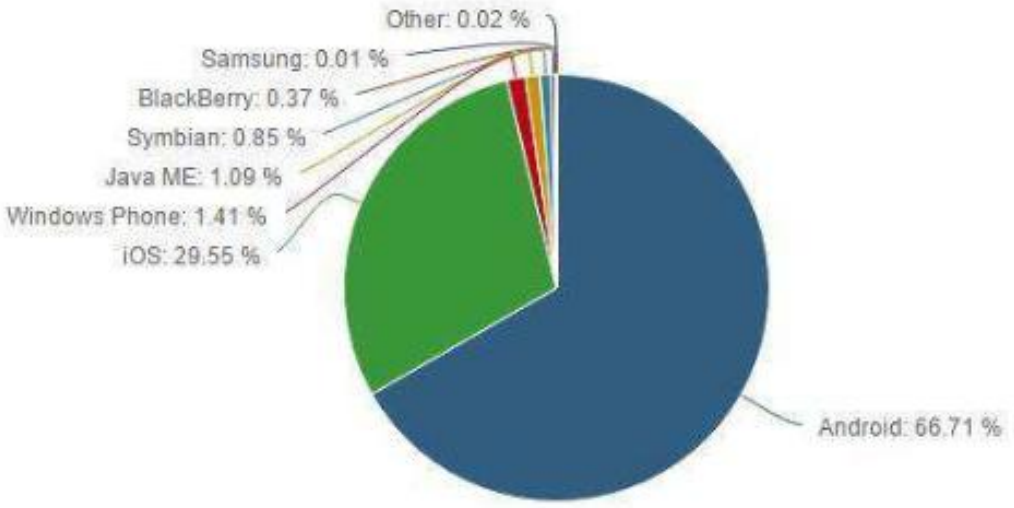

Рис. 1.1 - Діаграма співвідношення мобільних ОС на ринку смартфонів.

Розглянемо більш детальніше найвідоміші за кількістю користувачів операційні системи, а саме:

- Windows Phone

- iOS

- Android

Windows Phone

Windows Phone мобільна операційна система, розроблена американською компанією Microsoft. Вона встановлена на 1,41% смартфонів. Через невеликої частки ринку і неясних перспектив число додатків для Windows Phone помітно поступається кількості додатків для Android і iOS, проте магазин додатків Windows Phone Store може задовольнити практично будь-які потреби, тому що кількість додатків в ньому перевищує 300 тисяч.

Однак, з огляду на щорічне падіння популярності, розробка програми під дану ОС не є привабливою перспективою, так як охоплення користувачів буде вкрай малий.

iOS

iOS - операційна система для смартфонів, електронних планшетів і носяться програвачів, розроблена американською компанією Apple. Частка iOS на ринку мобільних операційних систем (29,55%) в точності відображає частку ринку айфонів ринку смартфонів, на Т.К. встановлюється тільки на айфонах і айпедов (мобільних пристроях від компанії Apple), в той час як Android і Windows Phone використовуються різними виробниками смартфонів. Кількість додатків для 1OS в магазині додатків App Store перевищує мільйон. Дана операційна система є дуже привабливою для розробки, так як має велику аудиторію користувачів. Але, незважаючи на це, є ряд недоліків:

-кількість користувачів iOS все ж менше, ніж користувачів Android;

-недешевий аккаунт розробника для завантаження програми в App Store;

-відсутність необхідних інструментів для розробки.

Android

Android операційна система для смартфонів і безлічі інших пристроїв. Спочатку розроблялася каліфорнійською компанією Android Inc., яку потім купив американський пошуковий гігант Google. Частка Android на ринку ОС складає 66,71. Кількість додатків для Андроїд в магазині додатків Google Play перевищує 1,43 млн. Ця операційна має найбільшу аудиторію система користувачів, а, отже, як було сказано вище, приверне більший відгук серед встановили додаток. Аккаунт розробника для публікації додатка в Play Маркет теж платний, проте коштує набагато менше ніж в App Store і платіж здійснюється одноразово (в App Store щорічний платіж). Проаналізувавши популярні мобільні самі операційні системи було вирішено, що розробка

буде виконуватися під пристрої з ОС Android, так як вона найбільш соответствует заявленим вимогам.

1.3 Детальніше про систему Android

Android - операційна система для смартфонів, планшетів, електронних книг, цифрових програвачів, наручних годинників, фітнес-браслетів, ігрових приставок, ноутбуків, нетбуків, смартбуків, окулярів Google Glass, телевізорів та інших пристроїв (в 2015 році з'явилася підтримка автомобільних розважальних систем і побутових роботів).

Заснована на ядрі Linux і власної реалізації віртуальної машини Java від Google. Спочатку розроблялася компанією Android, Inc., яку потім купила Google. Згодом Google ініціювала створення альянсу Open Handset Alliance (OHA), який зараз займається підтримкою і подальшим розвитком платформи. Android дозволяє створювати Java-додатка, що керують пристроєм через розроблені Google бібліотеки. Android Native Development Kit дозволяє перенести бібліотеки і компоненти додатків, написані на Ci та інших мовах.

У 86% смартфонів, проданих в усьому світі у другому кварталі 2014 року, була встановлена операційна система Android. На конференції для розробників травнем 2017 року Google оголосила, що за всю історію Android було активовано більше 2 млрд Android-пристроїв. Карти Google (GoogleMaps) - набір додатків, побудованих на основі безкоштовного картографічного сервісу і технології, що надаються компанією Google створених в 2005 році.

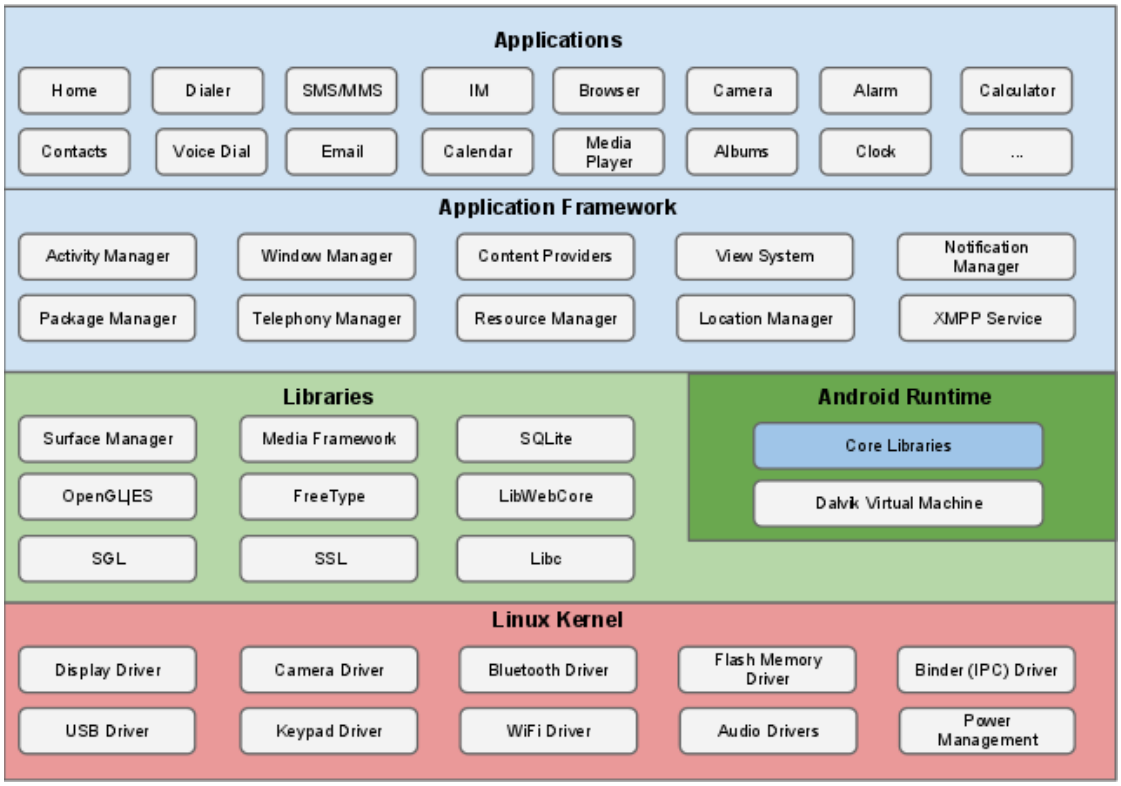

Рис. 1.2 - Програмний стек ОС Android

Цікавою рисою операційної системи Android є той факт, що додаток в ній може використовувати для свого функціоналу елементи інших додатків, якщо, звичайно, вони надають такий доступ. Наприклад, якщо нашому додатку потрібно відобразити список зображень, а інший додаток вже має реалізований відповідний скроллер, відкритий для інших додатків, то можна просто викликати його для здійснення цієї дії і не розробляти свій власний. При цьому код нашого додатки не змішується з кодом іншого і не компонується з ним. Швидше, він просто запускає деякий шматок іншого файлу, коли виникає така необхідність. Для забезпечення такого принципу роботи, система повинна бути здатна запускати процес додатки тоді, коли буде потрібно будь-яка його частина і створювати екземпляри java-об'єктів саме для цієї частини. Тому додатки Android не мають єдиної точки входу, як це прийнято в більшості систем. Замість цього їх код являє собою набір

деяких окремих цілісних сутностей, компонент, з яких система в міру необхідності може створювати екземпляри і використовувати їх.

Можна виділити чотири типи таких сутностей-компонент. Поговоримо про кожну окремо. Являє собою зовнішній інтерфейс для однієї операції, яку може зробити користувач. Якщо спростити, то це просто один поточний екран як деяка одиниця активності, свого роду кадр з одним призначеним для користувача дією. Тут і далі буде використовуватися термін activity без перекладу, як деякий власна назва. Хоча дослівний переклад, «активність» або «дія», добре передає загальний зміст компонента.

Наприклад, activity може надати список пунктів меню, які може вибрати користувач або відобразити фотографії з їх підписами. Або інший приклад - додаток для миттєвого обміну повідомленнями може використовувати одне activity для того щоб відобразити лист контактів, інше - щоб створити повідомлення для обраного контакту, третє - щоб подивитися історію повідомлень або виконати налаштування і так далі.

Все activity поточного додатка працюють разом і формують єдиний призначений для користувача інтерфейс, проте при цьому вони незалежні між собою. Кожне з них реалізовано як підклас базового класу Activity, що забезпечує створення вікна, в якому програміст може помістити візуальний інтерфейс.

Додаток може складатися з усього одного activity або відразу з декількох, як згаданий раніше в якості прикладу месенджер. Якими саме будуть activity і скільки їх буде, залежить від конкретного додатка і його дизайну. Як правило, одне з activity позначається як перше, це означає, що воно буде надано користувачеві під час запуску програми. Одне activity може запускати інше. Таким чином, перехід від одного activity до іншого здійснюється тоді, коли поточний activity викликає наступне.

Кожне activity надає вікно за умовчанням. Зазвичай вікно створюється в повноекранному вигляді, але воно також може і не займати весь екран і перебувати поверх інших вікон. Activity також може задіяти додаткові вікна - наприклад, спливаюче діалогове для взаємодії з користувачем в процесі роботи activity, або вікно для надання поточної інформації при виборі будь-якої важливої опції.

Візуальне зміст вікна будується за допомогою ієрархії візуальних компонентів (або уявлень) - об'єктів, похідних від базового класу View. Кожен компонент являє собою просто прямокутний простір всередині вікна. Батьківські компоненти містять дочірні і організують їх розташування. Ієрархію компонент можна представити у вигляді дерева, а ті елементи, які знаходяться в самій нижній його частині (листя) і не мають дочірніх компонент, описують прямокутні області і очікують дій користувача на цій ділянці. Таким чином здійснюється інтерактивне взаємодія з користувачем. Наприклад, таке подання може відображати на екрані маленьку іконку і ініціювати будь-яка дія, коли користувач на неї натисне. В операційній системі Android вже є набір готових візуальних компонент, які доступні для використання розробниками. Набір включає в себе кнопки, текстові поля, смуги прокрутки, меню, прапорці-перемикачі і багато іншого.

Для того щоб помістити у вікно таку ієрархію, потрібно викликати метод Activity.setContentView (). Параметром методу є екземпляр класу View, що лежить в корені ієрархії. (Сервіси) являють собою компоненти, які працюють у фоновому режимі. Він, як правило, потрібно для тривалих операцій або для забезпечення роботи віддалених процесів, але в загальному випадку це просто режим, який функціонує, коли додаток не в фокусі. Прикладом такого процесу може стати прослуховування музики в той час, коли користувач робить щось інше або отримання даних по мережі без блокування поточної активності. Сервіс сам по собі не надає призначеного для користувача інтерфейсу, тобто з користувачем не взаємодіє, а

запускається, управляється і пов'язаний з іншими компонентами, наприклад, activity. Також може запускатися разом з системою providers. Даний компонент управляє наборами даних, які додатки надають іншим. Ці дані можуть зберігатися в файлової системі, базах даних SQLite, в мережі, або в будь-якому іншому постійному місці, до якого додаток може мати доступ. За допомогою content provider інший додаток може запитувати дані і, якщо виставлені відповідні дозволи, змінювати їх.

У більш загальному випадку, content provider можна використовувати для читання і запису даних, які використовуються додатком і не є відкритими для інших. Наприклад, додаток NotePad використовує такий компонент для збереження зроблених записів.

Дані компоненти реалізуються як підклас ContentProvider. І для того, щоб інші програми могли зробити операції з даними, їм необхідно надати стандартний набір API.receivers. Цей компонент відповідає за поширення загальносистемних повідомлень, відстеження та реагування на дії. Багато оповіщення йдуть від системи, наприклад, повідомлення про те що заряд батареї малий або екран вимкнений. Додатки також можуть ініціювати такі оповіщення, наприклад, сигналізувати про те, що інформація завантажена на пристрій і доступна до використання. Як і сервіси, broadcast receiver не надає призначеного для користувача інтерфейсу, проте, він здатний створювати повідомлення в рядку стану, щоб отримувати повідомлення про те, що сталося якусь подію. Однак частіше broadcast receiver взаємодіє з іншими компонентами для того, щоб самому виконувати мінімальний обсяг роботи. Так, він може ініціювати сервіси для виконання дій, прив'язаних до якоїсь події.

1.4 Вибір середовища розробки

Найпопулярнішими середовищами розробки під операційну систему Android є:

- Eclipse

- Microsoft Xamarin

- Android Studio

- Microsoft Visual Studio

Розглянемо кожну з них

Eclipse

Eclipse є безкоштовною програмною платформою з відкритим вихідним кодом, контролюється організацією Eclipse Foundation. Написана на мові програмування Java і основною метою її створення є підвищення продуктивності процесу розробки програмного забезпечення. Eclipse сама по собі не є середовищем розробки додатків для мобільних пристроїв, але до неї можна підключити окремий плагін ADT (Android Development Tools).

Тим не менш, дана середовище розробки не орієнтована саме на розробку мобільних додатків, тому можливі різні проблеми на етапі розробки, яких можна уникнути, вибравши інше середовище розробки.

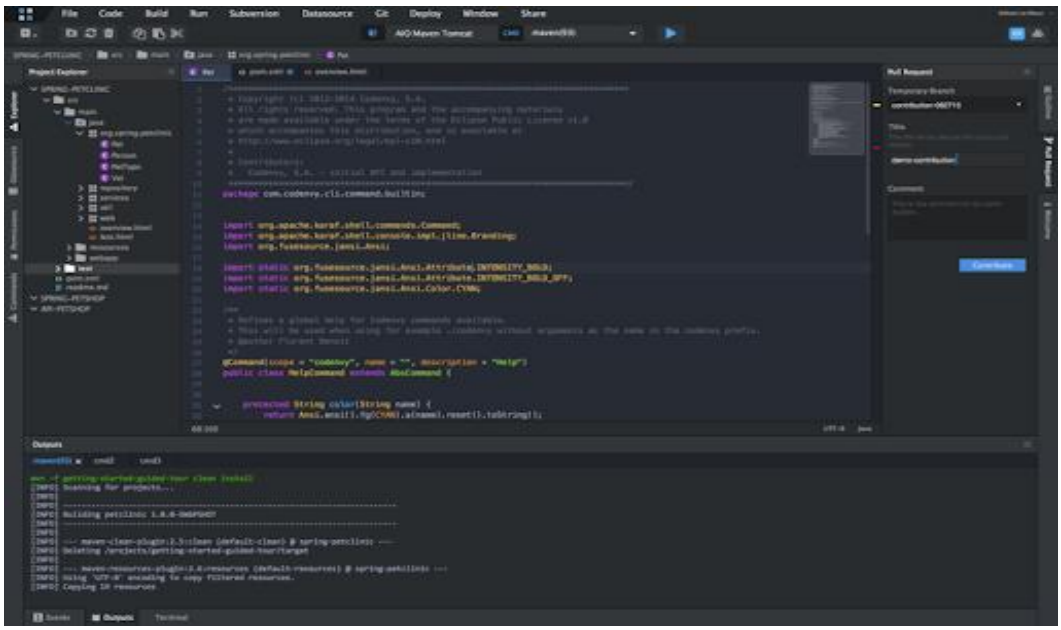

Рис. 1.3 – Середа розробки Eclipse

Microsoft Xamarin

Microsoft Xamarin це платформа розробки мобільних додатків для створення нативних додатків iOS, Android і Windows із загального коду C # або NET, яка дозволяє багаторазово використовувати між платформами від 75% до майже 100% коду. Програми, написані за допомогою Xamarin і C #, мають повний доступ до інтерфейсів API базової платформи і можливість створювати нативні призначені для користувача інтерфейси, а також компілювати код в машинний, тому вплив на продуктивність під час виконання є незначним.

Розробники, знайомі з C #, NET і Visual Studio, можуть розраховувати на такі ж можливості і продуктивність при роботі з Xamarin для мобільних додатків, включаючи віддалену налагодження на пристроях Android, iOS і Windows, без необхідності вивчати нативні мови, наприклад, Objective-C Java. Дивно, чи але багато високопродуктивних додатків з красивими користувача інтерфейсами наприклад, NASCAR, Aviva і MixRadio створені допомогою Xamarin.

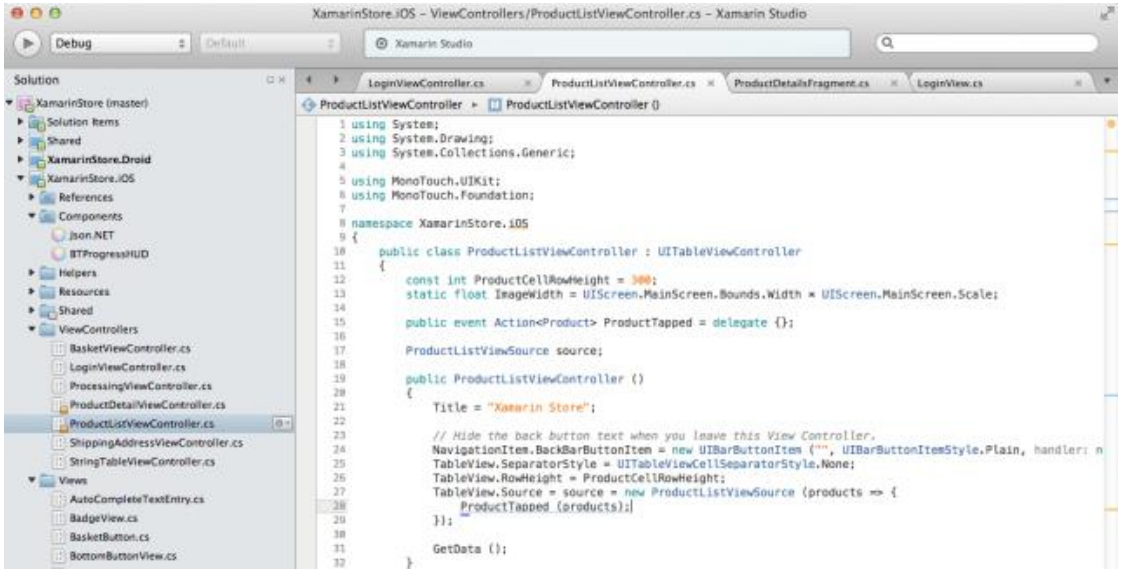

Рис 1.4 – Середа Microsoft Xamarin

Android Studio

Android Studio це інтегроване середовище розробки (IDE) для роботи з платформою Android, анонсована 16 травня 2013 року на конференції Google IO [6]. Це молода середовище розробки, але вже дуже популярна серед розробників під ОС Android. Вона ряд має позитивних особливостей. Ось деякі з них:

- розширений редактор макетів: WYSIWYG, здатність працювати з U1 компонентами за допомогою Drag-and-Drop, функція попереднього перегляду макета на декількох конфігураціях екрану;

- збірка програм, заснована на Gradle;

- різні види збірок і генерація кількох apk файлів;

- рефакторинг коду;

- статичний аналізатор коду (Lint), що дозволяє знаходити проблеми продуктивності, несумісності версій і інше;

- шаблони основних макетів і компонентів Android;

- підтримка розробки додатків для Android Wear і Android TV;

- Android Studio 2.1 підтримує Android N Preview SDK, а це значить, що розробники зможуть почати роботу зі створення програми для нової програмної платформи.

Також, дана середовище розробки, як випливає з назви, орієнтована на розробку додатків саме під ОС Android, що є істотною перевагою перед іншими IDE.

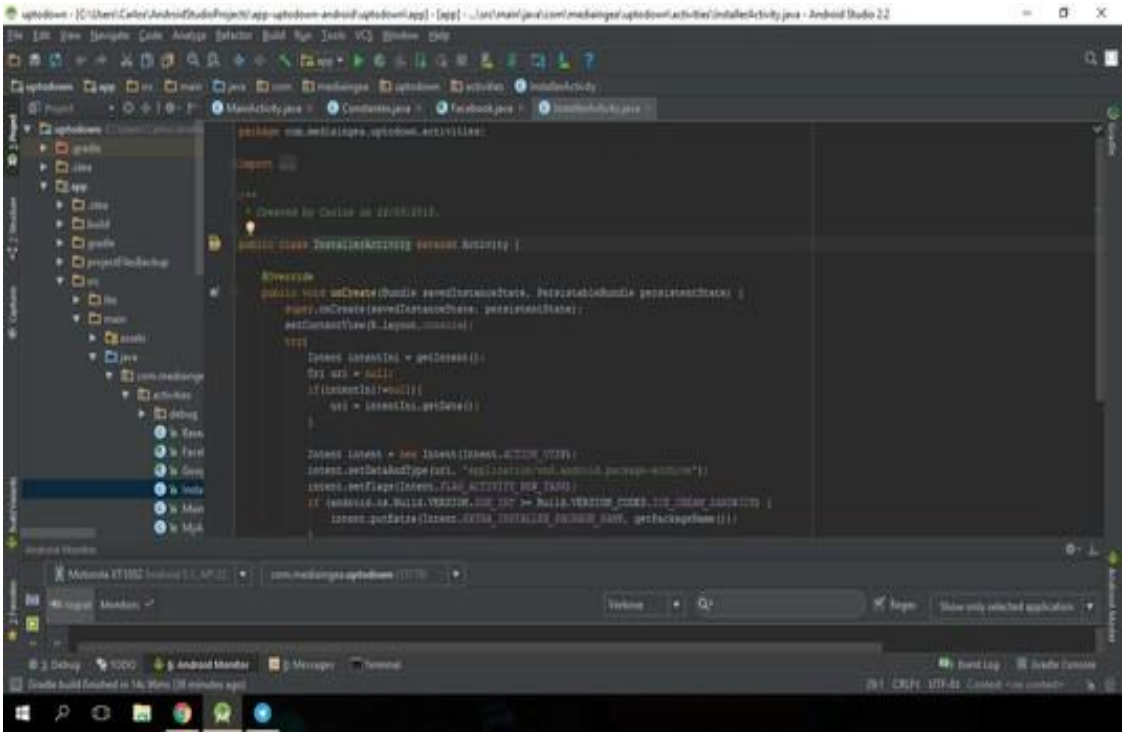

Рис. 1.5 – Зовнішній вигляд Android Studio.

Microsoft Visual Studio

Microsoft Visual Studio - лінійка продуктів компанії Microsoft, що включають інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-додатки, веб-

служби як в рідному, так і в керованому кодах для всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight.

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Решта вбудовуються інструменти включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server).

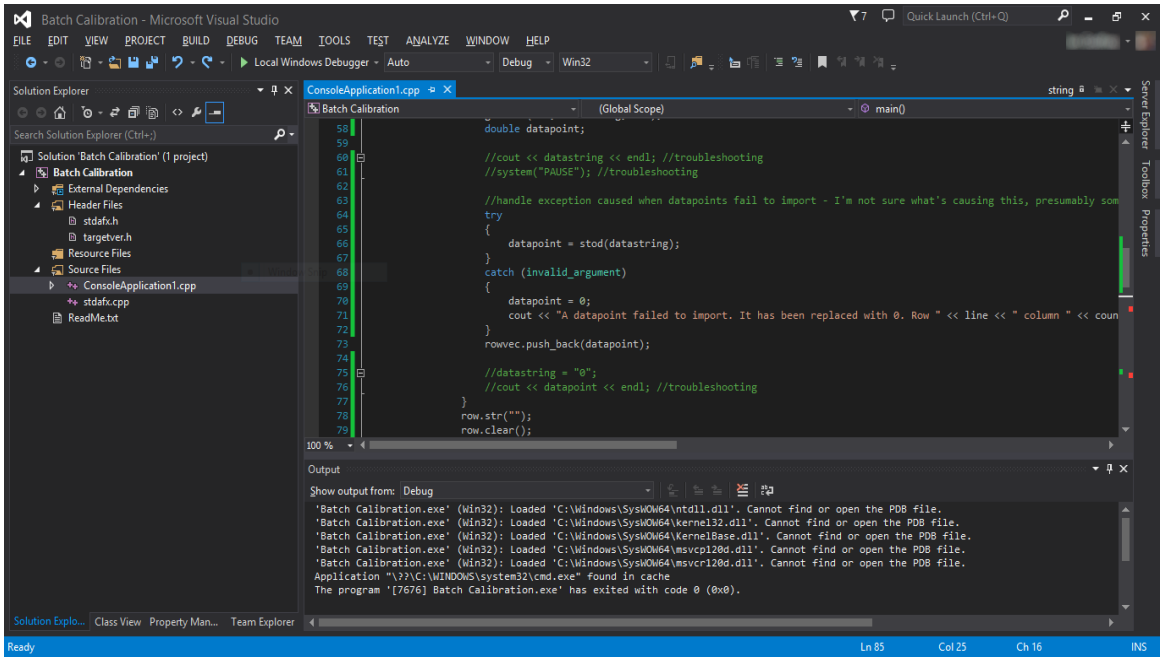

Рис. 1.6 – Середа розробки Microsoft Visual Studio

Ключові особливості Android

Операційна система Android досить унікальна. Для отримання хорошого результату розробнику може знадобитись знаючи багато тонкощів і особливостей даної операційної системи. При розробці існують кілька труднощів, які важливо враховувати. Таких як:

Додаток після установки вимагає в два рази, а в деяких випадках навіть в чотири рази більше місця, ніж розмір оригінального програми, На вбудованої флеш-карті швидкість роботи з фалами падає в кілька десятків разів при невеликій кількості вільного місця, Процеси можуть використовувати до 16 Мб , а деяких випадках і 24 Мб оперативної пам'яті. Між ядрами і додатком лежить свій шар API і на нативному коді-шар бібліотек. В Android можна запускати необмежену кількість додатків, яке дозволяє оперативна пам'ять в пристрої. Але тільки один з додатків є головним і відображається на екрані. Від відкритої програми можна перейти до фонового режиму або запустити нове. Даний процес візуально нагадує вкладки браузера. Екрани інтерфейсу користувача представлені класом Activity в коді і містяться в процесах. Activity може жити довше процесу. Activity при зупинці може бути запущено знову в збереженням всієї потрібної нам інформації. При цьому використовуються спеціальні механізми опису дій засновані на Intent. Когда потрібно зробити дзвінок, надіслати листа, показати вікно або виконати дії, викликається Intent.

Також Android містять задану сервіси подібні демонам в Linux для виконання потребовавшихся дій у фоновому режимі наприклад, як програвання музики або відео. В таких цілях використовується Content providers (провайдери вмісту) для обміну даними між додатками. Загальна схема робота додатків Android Програми для Android використовують вікна в своїй роботі подібно Windows вікон, хоча в даній системі вікна мають іншу назву - Activity. Як і в Windows, будь-яке вікно має свій життєвий цикл і свої

характерні риси. Схема представляє життєвий цикл програми для Android зображена на рис. 1.3

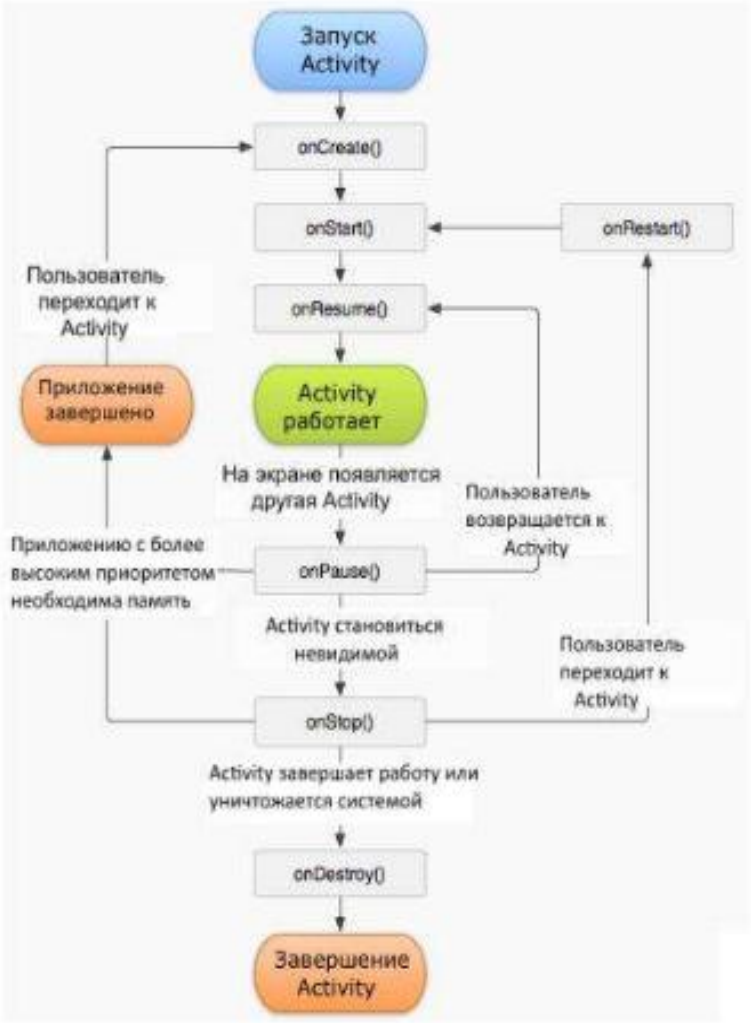

Рис. 1.7 – Життєвий цикл програми для Android

Для того щоб створити нове вікно викликається метод onCreate (), при розробки цей метод перепризначувався і в ньому відбувається ініціювання програми та його елементів. Далі слід метод onStart) і метод onResume (). Які визиваються перед показом онка при його створенні, або його відновлення (при переході з іншої програми, при відкритті згорнутого додатки і так далі). При згортанні використовується метод onDestroy (), в даному методі зберігаються параметри і дані поз ователя. Щоб отримати послідовність викликів методів і більш повний опис їх можна знайти на офіційному сайті.

1.5 Інструменти розробки

Безпосередньо перехід до практики програмування під будь-яку платформу починається з розгляду наявного арсеналу інструментів розробки. Уже після того, як інструменти обрані (керуючись якимись міркуваннями і уподобаннями), налаштовані для роботи і випробувані в написанні яких-небудь простих інструкцій, починається «велика» програмування та вдосконалення навичок.

Перш ніж розглядати інструменти для Android, необхідно ввести деякі загальні поняття, з якими доведеться зіткнутися на даному етапі. Перше базове поняття - це SDK або Software Development Kit («devkit»). Під ним розуміють набір засобів розробки, що дозволяє програмістам створювати додатки під деяку архітектуру - для певного пакету програм, апаратної або програмної платформи, операційної або комп'ютерної систем. SDK, як правило, надає розробнику широкий набір інструментів, який може включати в себе засоби налагодження і довідкові матеріали (приклади коду, замітки і т.д.). SDK часто надається безпосередньо розробником цільової платформи і в ньому враховуються її особливості. Поширяться (як і у випадку з Android) такий інструментарій може безкоштовно.

Ще одним поняттям, яке часто можна зустріти, коли мова йде про інструменти розробки, є IDE (Integrated development environment) - інтегроване середовище розробки. Вона являє собою набір програмних засобів, покликаний максимізувати продуктивність програміста за рахунок побудови для нього зручною і доброзичливою середовища для процесу програмування. Цей складний програмний комплекс може включати в себе текстовий редактор (найчастіше з підсвічуванням синтаксису і підказками),

компілятор або інтерпретатор (іноді підтримуються кілька мов), інтегрований відладчик і засоби для автоматизації збирання. Часто підтримуються можливості інтеграції зі стороннім програмним забезпеченням, наприклад, засобами проектування або контролю версій. Також найчастіше в IDE є інструменти для швидкого побудови і візуального редагування графічного інтерфейсу майбутньої програми, а також інструменти на зразок діаграм ієрархії класів, браузера класів, інспектора об'єктів або менеджера ресурсів.

Таким чином, інтегроване середовище розробки покликана об'єднати різні інструменти в один програмний комплекс для забезпечення зручності і підвищення швидкості розробки ПО. Хоча на ділі можна і не використовувати для програмування будь-яку IDE, вважаючи за краще незв'язні між собою інструменти. інтегроване середовище розробки покликана об'єднати різні інструменти в один програмний комплекс для забезпечення зручності і підвищення швидкості розробки ПО. Хоча на ділі можна і не використовувати для програмування будь-яку IDE, вважаючи за краще незв'язні між собою інструменти. інтегроване середовище розробки покликана об'єднати різні інструменти в один програмний комплекс для забезпечення зручності і підвищення швидкості розробки ПО. Хоча на ділі можна і не використовувати для програмування будь-яку IDE, вважаючи за краще незв'язні між собою інструменти.

Розробка додатків для платформи Android пов'язана з групою інструментів, які надаються набором Android SDK. Також знадобляться інструментарій для розробки додатків в Java SE (JDK) і інтегроване середовище розробки. В якості останньої прийнято використовувати Eclipse IDE, що будемо робити і ми, хоча варто відзначити, що розробку додатків можна вести і за допомогою найпростішого текстового редактора або інших IDE, а також викликати інструменти за допомогою скриптів або використання командного рядка. Однак розробка в Eclipse є найкращим методом, тому що, по-перше, це середовище може безпосередньо звертатися

до необхідних інструментів, а по-друге, для неї існує спеціальний плагін, Android Development Toolkit (ADT),

У висновку відзначимо, що Android SDK вимагає JDK версії 5 або вище, а також Eclipse версії 3.3 або вище. На сайтах розробників доступні версії SDK, Java і Eclipse для операційних систем Windows, Linux і Mac OS, а в SDK входить емулятор для кожної з них. Взагалі кажучи, додатки під Android виконуються у віртуальній машині, і вибір будь-якої з існуючих операційних систем не дає ніяких переваг перед іншими, так що можна працювати з тією, яка більш зручна. Наші подальші приклади будуть розглянуті для ОС Windows, але, якщо буде необхідність, ми розглянемо і роботу з Linux.

Для створення програм на мові Java необхідне спеціальне програмне забезпечення. Найостанніші версії цього ПО можна завантажити з офіційного сайту розробника, Oracle Corporation.

До цього програмного комплексу відносяться такі інструменти як JRE (Java Runtime Environment) і JDK (Java Development Kit). Перший інструмент являє собою середовище виконання - мінімальну реалізацію віртуальної машини, в якій запускається і виконується програмний код на Java. Другий інструмент - це в свою чергу цілий набір інструментів, комплект розробника додатків на мові Java. Насправді, JRE також входить до складу JDK, так само як і різні стандартні бібліотеки класів Java, компілятор javac, документація, приклади коду і різноманітні службові утиліти. Весь цей набір поширюється вільно і має версії для різних ОС, тому будь-хто може його скачати і використовувати.

В JDK не входить інтегроване середовище розробки, передбачається, що її розробник буде встановлювати окремо. Існують численні IDE для Java-розробки, наприклад, NetBeans, IntelliJ IDEA, Borland JBuilder і інші. Але ми обумовили раніше, що для розробки додатків під Android ми виберемо

Eclipse IDE. Для установки JDK, необхідно спочатку завантажити її з сайту розробника <# "justify"> o.

Якщо раптом під час розпакування архіву (у випадку з exe-файлом) з'являються повідомлення про відсутність в системі встановленого JDK (а він при цьому встановлений), то досить натиснути кнопку Back (Назад) і знову спробувати, тоді все стане коректно.

Після виконаних дій ми підемо в папку з розпакованим контентом і почитаємо SDK Readme.txt, в якому йдеться про те, що цей архів містить в собі тільки базові інструментальні засоби. Для повноцінної розробки нам належить скористатися утилітою SDK Manager, яка дозволяє встановлювати і змінювати компоненти SDK, користуючись репозиторієм на сервері Google.

Запустимо виконуваний файл з назвою SDK Manager. Тут може виникнути проблема з його роботою. Якщо видаються помилки і програма не запускається, то тут справа в змінного середовища PATH, яка зберігає в собі шляхи пошуку для виконуваних файлів і використовується різними програмами і скриптами. Для коректної роботи в цій змінній повинен бути вказаний шлях до каталогу інструментів Android SDK, а також до каталогу bin для JDK.

РОЗДІЛ 2 ПІДГОТОВКА ДО РЕАЛІЗАЦІЇ

Исходник буде виконаний у вигляді повноцінного додатка для операційної системи Android, у вигляді книги для читання. Компонування елементів побудована на контейнері TableLayout. Комбінування різних типів дочірніх об'єктів всередині TableLayout забезпечило можливість об'єднання осередків як по рядках, так і по стовпцях. При дотику (асоціація з кліком кнопки миші OnClick) до радіокнопку з назвою байки, в візуальний елемент TextView завантажується відповідний текст. При кліки на радіокнопку налаштувань можна змінювати колір фону і розмір шрифту. Довгий дотик до колонку з текстом приховує або показує стовпець налаштувань.

Розмітка TableLayout (Табличная розмітка) позиціонує свої дочірні елементи в рядки і стовпці, як це звикли робити веб-майстри в тезі table. TableLayout не відображує лінії обрамлення для їх рядків, стовпців або комірок. TableLayout може мати рядки з різною кількістю осередків. При формуванні розмітки таблиці деякі осередки при необхідності можна залишати порожніми. При створенні розмітки для рядків використовуються об'єкти TableRow, які є дочірніми класами TableLayout (кожен TableRow визначає єдиний рядок в таблиці). Рядок може не мати осередків або мати одну і більше осередків, які є контейнерами для інших об'єктів. У осередок допускається вкладати інший TableLayout або LinearLayout.

TableLayout зручно використовувати, наприклад, при створенні логічних ігор типу Судоку, Хрестики-нулики і т.п.

Ось кілька правил для TableLayout. По-перше, ширина кожної колонки визначається по найбільш широкому вмісту в колонці. Дочірні елементи використовують в атрибутах значення match_parent. Атрибут TableRow для layout_height завжди wrap_content. Осередки можуть об'єднувати колонки, але не ряди. Досягається злиття колонок через атрибут layout_span.

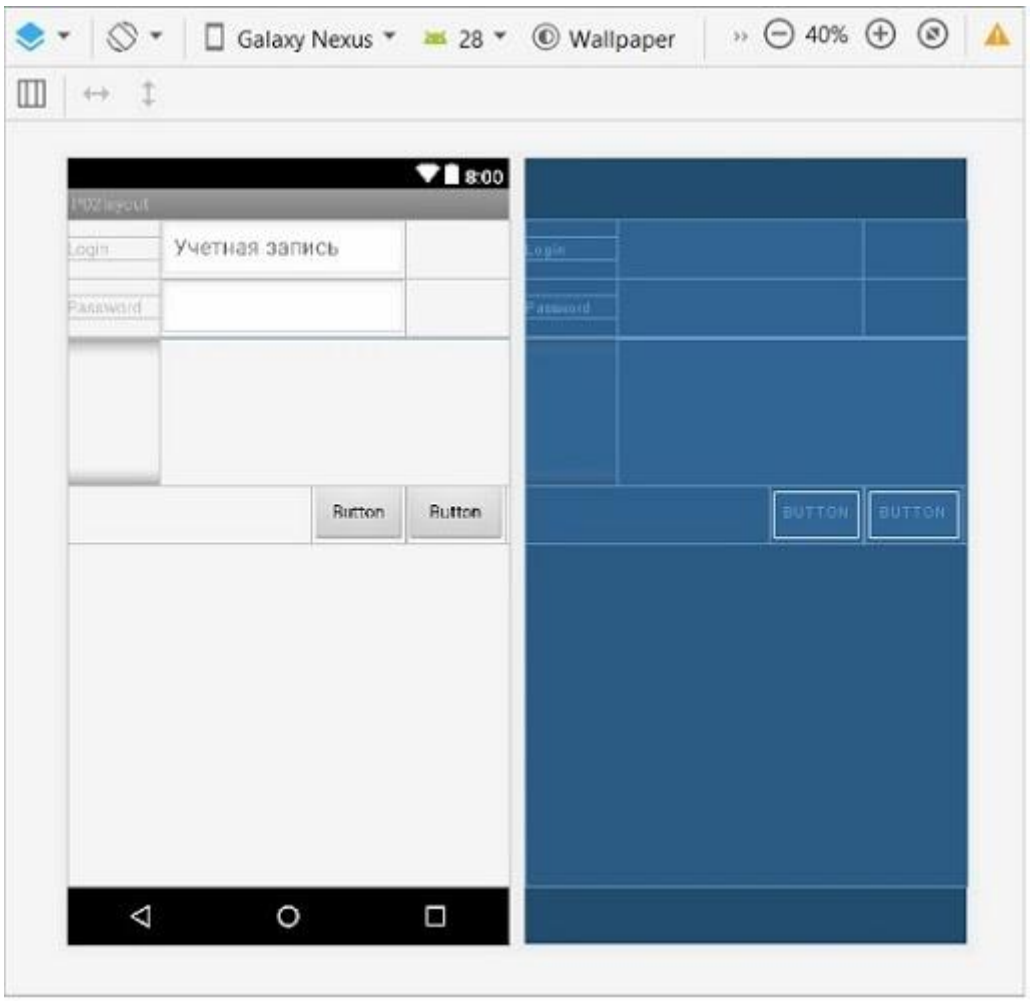

Рис. 2.1 – Зовнішній вигляд TableLayout

2.1 Макет TableLayout

Інтерфейс програми побудований на макеті Android TableLayout. Даний макет зручний для створення інтерфейсу програми на основі строгих осередків, утворених рядками TableRow. Стовпці макета існують тільки в межах рядка і не можуть бути об'єднані по вертикалі з осередками інших рядків. TableLayout нагадує віддалено таблицю html з тегом table. Але веб таблиця допускає об'єднання осередків по рядках і по стовпцях. Tablelayout ж допускає об'єднання осередків тільки всередині рядка. Рядки об'єднувати, можливо поки, не можна. Але є і гідність цих властивостей макета TableLayout: всередині рядка комірки можуть бути будь-якої бажаної ширини. Крім того, для обходу неможливості об'єднання рядків можливе

застосування в якості елементів таблиці сам же макет Tablelayout і інші контейнери.

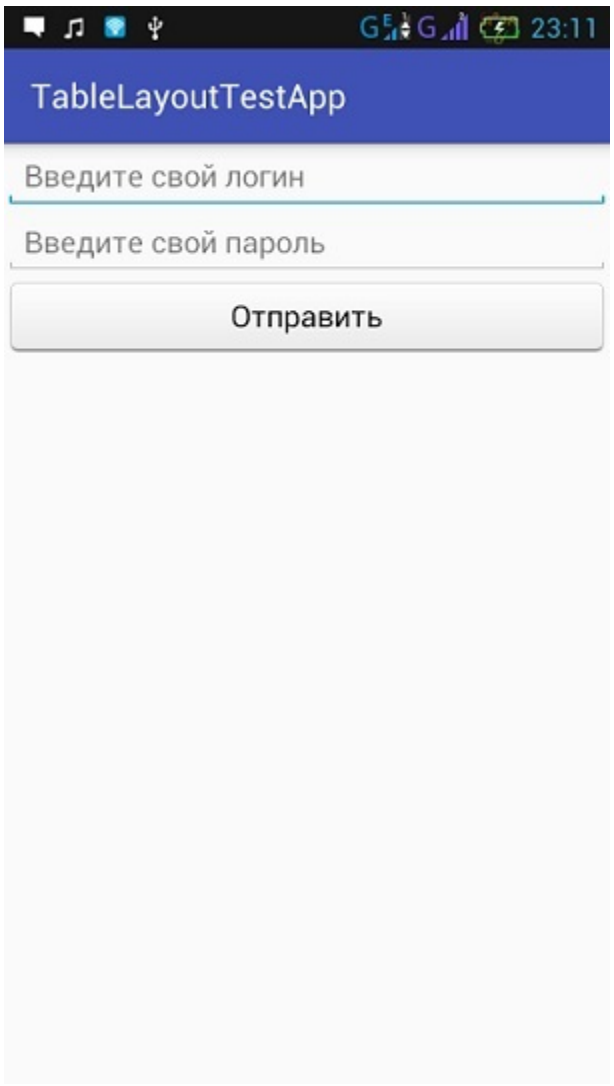

Рис. 2.2 – Макет TableLayout

2.2 Елементи TableLayout

Для визначення рядка в макеті TableLayout стандартно призначені об'єкти TableRow. У TableRow можна додавати будь-які візуальні і контейнерні елементи. Простір між дочірніми елементами розподіляється абсолютними і відносними величинами. Абсолютними безпосередньо вказується ширина і висота, відносні розподіляють простір на основі ваг

layout_weight. Комбінуючи ці параметри, досягається бажана компоновка рядків і осередків. У TableLayout, минаючи вставку TableRow, як рядків можна додавати будь-який вид і контейнер, наприклад: TextView, RadioGroup і навіть сам контейнер TableLayout. Але тільки TableRow формують осередки всередині рядка. Будь-які інші види елементів створюють осередок на всю ширину материнського контейнера. Додаючи наступні види, всі вони будуть займати всю доступну ширину.

Об'єкт netAlert - текстове поле, у якому з'являється попередження, у разі виникнення проблем підключення до мережі. Для того, щоб перевірити стан підключення, потрібно отримати об'єкт ConnectivityManager. Він відповідає за запити щодо стану мережі та інформує програму, коли стан мережі змінюється. У цього об'єкту є метод, який в свою чергу повертає об'єкт NetworkInfo, в якому знаходиться інформація про підключення до мережі. Щоб дізнатись, чи пристрій підключений до мережі, потрібно перевірити, чи об'єкт не пустий, чи пристрій підключений до мережі та чи мережа доступна. За це відповідає метод private final boolean isOnline().

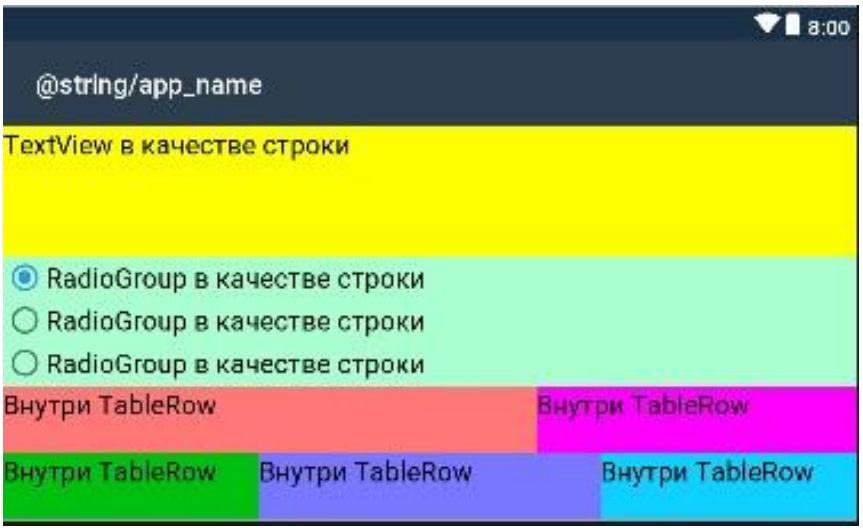

Рис. 2.3 - Елементи TableLayout

2.3 Компонування елементів

Перший рядок TableLayout в XML інтерфейсі исходника додатки це TextView як заголовок. Нижче доданий об'єкт TableRow дає можливість сформувати два осередки. Для забезпечення гарантованого перегляду всього змісту екрану програми цими осередками є елементи прокрутки ScrollView. Правий стовпчик складної компонування і містить безліч візуальних елементів. Так як в ScrollView можна додати тільки один дочірній елемент, то другий стовпець побудований на контейнері TableLayout. І в даному контейнері вже розміщується необхідну кількість елементів управління, що складається з TextView і безлічі RadioButtons управління.

Рис. 2.4 - Компонування елементів

2.4 Створення дизайну додатку

Додаток исходника має один вид екрану, одне Activity (вікно з інтерфейсом для користувача). Клас Activity визначає події пов'язані з життєвим циклом додатка і інтерактивністю інтерфейсу. Зовнішній вигляд

програми створюється частково в дизайнера XML і частково програмним способом. Дизайнер XML зручний тим, що дозволяє візуально сформувати компоновку додатки. Без програмного способу не обійтися, коли компоновка додатки формується динамічно на основі інформації, отриманої з бази даних. Комбінування дизайнерського і програмного способу дозволяє створювати динамічні інтерфейси додатків для операційної системи Android.

2.5 База даних програми

За базу даних виступає файл XML, що славиться своєю структурованістю і зручною можливістю прямого редагування. Файл складається з елементів Story які в свою чергу мають елементи назви і тексту розповіді, відповідно Name і Body. Програмний код зчитує всі елементи Story / Name і автоматично створює об'єкти RadioButtons з відповідною назвою. Це назва оповідання і є ключем виведення тексту на екран Android пристрою. Файл stories.xml знаходиться в каталозі Assets. Папка Assets спеціально призначена для додаткових ресурсів програми і ці ресурси доступні через сервіс Android AssetManager.

У файлі макету діяльності Start розташовані такі елементи: текстове поле, два поля для вводу даних (логіну та паролю), приховане текстове поле, у якому у разі потреби, з'являється попередження про відсутність доступу до мережі, та дві кнопки («Увійти» - у разі наявності аккаунта, «Створити» - у разі потреби створити новий аккаунт). Макет має відносне розташування (RelativeLayout). Розташування елементів задане у файлі стилів. Відповідні стилі підключені до кожного елементу макету.

Також даний файл містить меню. Меню працює за таким принципом: кожному елементу меню задаємо елемент Intent - об'єкт операції, яку потрібно виконати, та перевизначаємо метод, який відповідає за вибір деякого пункту меню. Після вибору пункту потрібно дістати об'єкт Intent з

вибраного елементу, та почати нову діяльність, передавши методу запуску діяльності (void startActivity (Intent intent)) цей об'єкт.

Щоб створити меню, потрібно перевизначити метод батьківського класу public boolean onCreateOptionsMenu (Menu menu) вказати макет меню, що знаходиться у папці menu, та присвоїти кожному елементу свою операцію

РОЗДІЛ 3 ПРОЕКТНА ЧАСТИНА

У даній роботі був реалізований мобільний додаток для перегляду текстових файлів. Нижче розглядається функціонал створеного додатку.

Головна задача цього додатку: відображення текстових файлів, у якості прикладу були завантажені байки І. А. Крилова.

Середовище програмування MS Visual Studio 2019. Цільова платформа API 26 (Android 8.0). Додаток тестувалося на смартфоні Android 8.0 версії.

Додаток підтримує мінімальну версію Android 4.0.4 і максимальну версію Android 9.0. Додаток добре оптимізований для підтримуваних версій незважаючи на використання елементів Material дизайн, який з'явився з виходом Android 5. На всіх рисунках продемонстровано роботу додатка на Android 8.0.

На рисунку 3.1 можна побачити загальний вигляд реалізованого додатку.

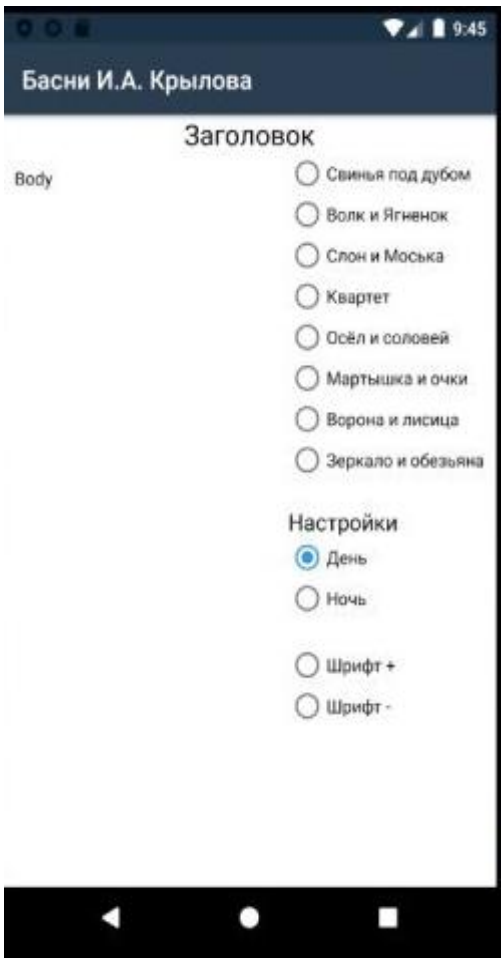

Рис. 3.1 – Загальний вид

У додатку реалізовано два режими перегляду: денний, нічний

У правій частині додатку можливо обрати один з завантажених текстів.

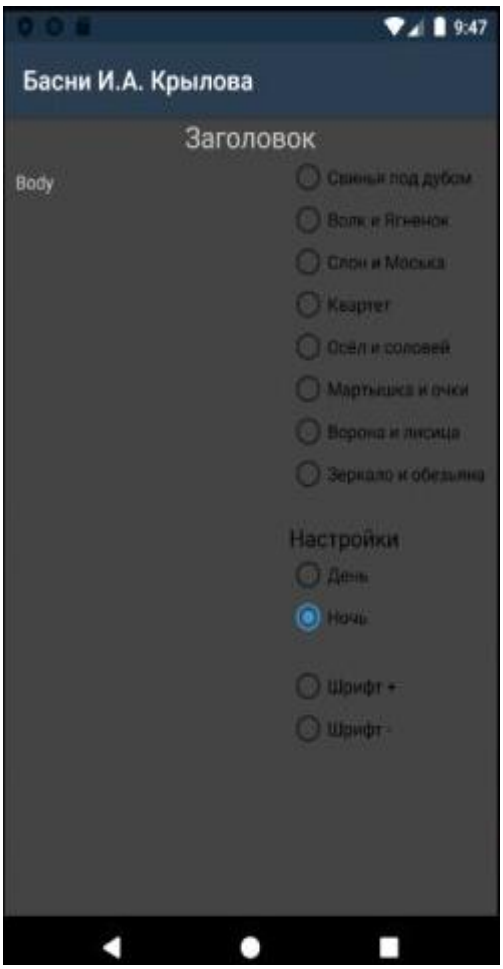

Рис. 3.2 – Нічний режим перегляду

У правій нижній частині екрану можливо редагувати налаштування зовнішнього вигляду додатку. На рисунку 3.2 ми можемо побачити нічний режим перегляду текстових файлів.

При дотику до кнопки з назвою байки, в візуальний елемент TextView завантажується відповідний текст. При кліках на кнопку налаштувань можна змінювати колір фону і розмір шрифту. Довгий дотик до колонку з текстом приховує або показує стовпець налаштувань.

На рисунку 3.3 бачимо денний режим перегляду.

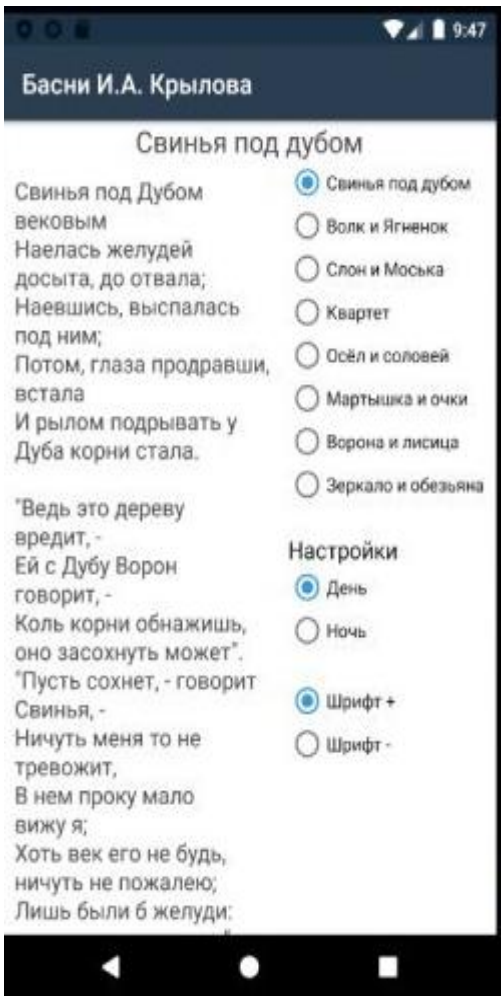

Рис. 3.3 – Режим «День» та «Шрифт +»

Також у додатку реалізована можливість змінювати розмір шрифту, на рисунку 3.4 можна побачити один з них: «Шрифт -»

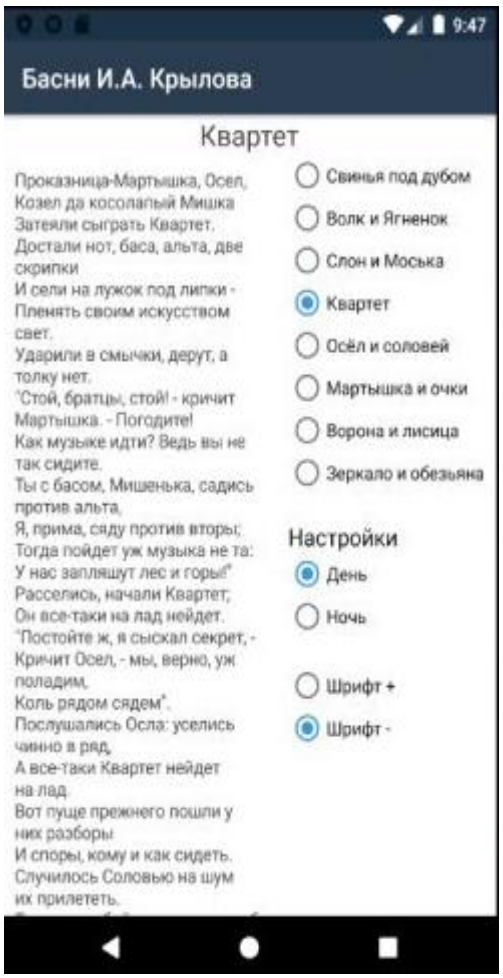

Рис. 3.4 – Режим «День» та «Шрифт -»

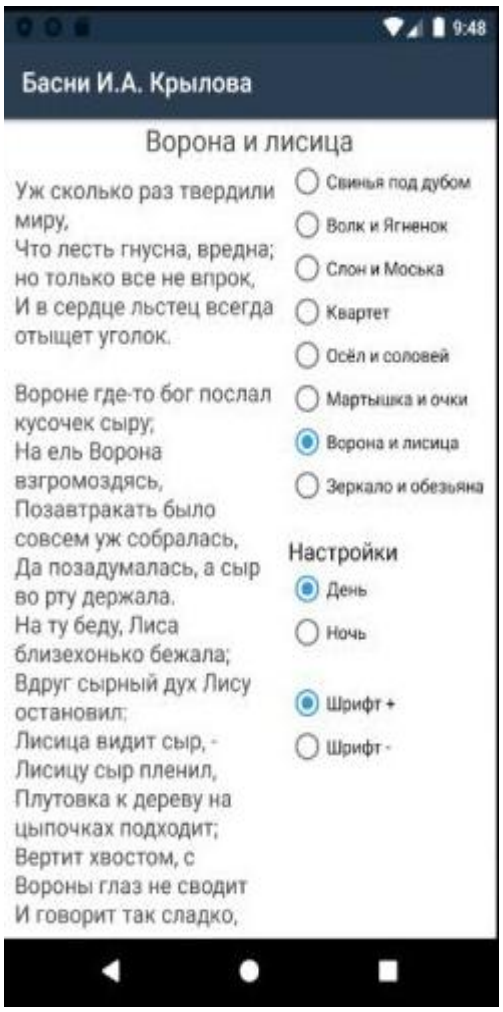

Рис. 3.5 – Режим «День» та «Шрифт +», перегляд текстового файлу

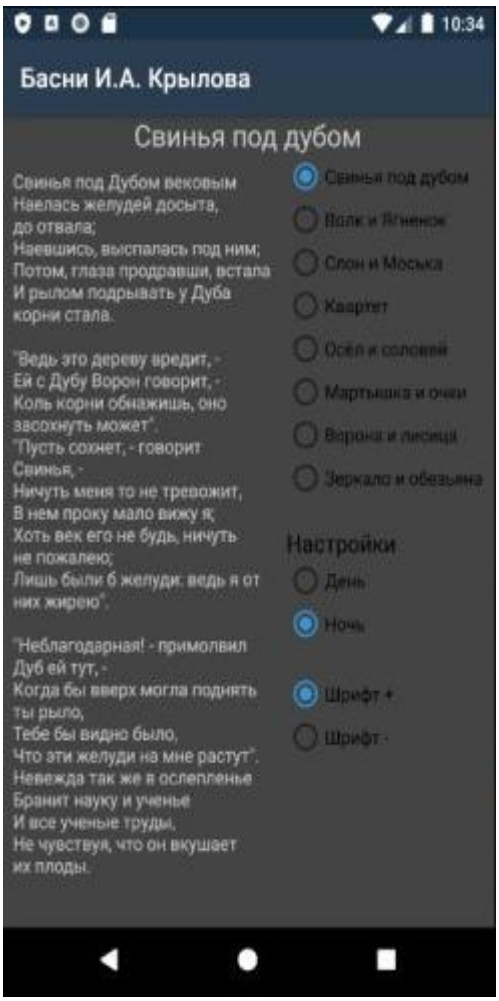

Рис. 3.6 – Режим «Ніч» та «Шрифт +», перегляд текстового файлу

Також додаток підтримує перегорнути вигляд(рис 3.7)

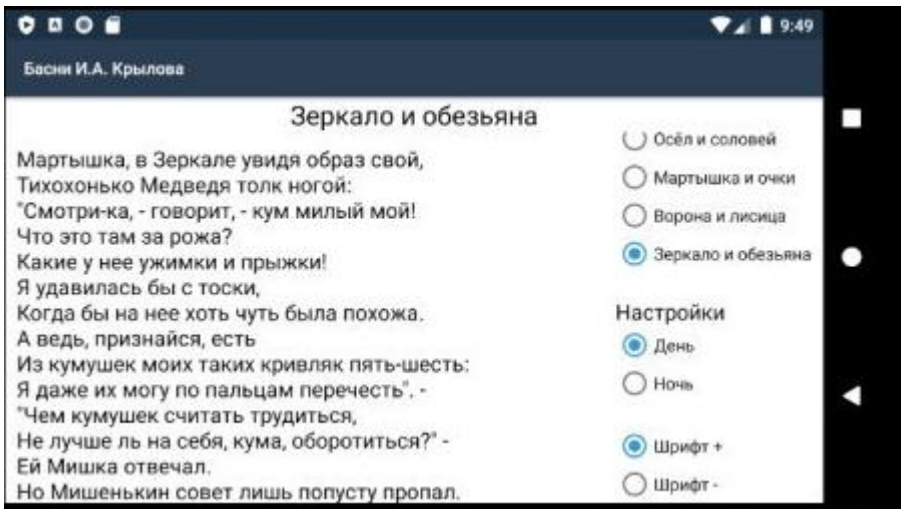

Рис. 3.7 – Режим «День» та «Шрифт +», перегорнутий вид

Код реалізації наведений у Додатку А.

ВИСНОВОК

У дипломній роботі було:

1) проведено аналіз предметної області;

2) вирішено розробити на базі каркаса Xamarin.Android для .NET. Xamarin.Android розширює платформу .NET додаючи можливість створювати додатки для Andriod на мові C #, F # і надає повний доступ до нативному (рідному) Android SDK;

3) обрано середовище програмування MS Visual Studio 2019;

4) та реалізований додаток.

СПИСОК ЛІТЕРАТУРИ

1. Мазалов В. В. Математическая теория игр и приложения : учеб. пособие / В. В. Мазалов. — Санкт–Петербург : Лань, 2010. — 448 с. — ISBN 978–5–8114–1025–5.2.
2. Стиллмен Э. Изучаем C# / Э. Стиллмен, Дж. Грин. — 3–е изд. — Санкт–Петербург : Питер, 2012. — 696 с. : ил. 3.
3. Хортон А. Microsoft Visual C++ 2005: базовый курс / Айвор Хортон. — Москва : Диалектика, 2007. — 1152 с. — ISBN 0–7645–7197–4.5.
4. Брюс Еккель. Філософія Java. «Питер» - 2006. - 648 с.
5. Офіційна документація операційної системи Android [Електронний ресурс]. Режим доступу до ресурсу: http://developer.android.com
6. Офіційна документація мови програмування Java [Електронний ресурс]. Режим доступу до ресурсу: http://docs.oracle.com/javase/6/docs
7. Вей-Менг Лі. Android Application Development Cookbook. «Питер» - 2013. -408 с.
8. Коматинени С., Маклин Д., Хашими С. Розробка програм для Android. «Питер». - 2011. -736 с.
9. Дейтел П., Дейтел Х., Дейтел Э., Моргано М. Android для програмістів. Створюємо програми на Android. «Питер». -2012. -560 с.

10 . Кращі операційні системи для смартфонів [Електронний ресурс] Режим доступу: http: //topmira.com/tehnika (http://to%D1%80mira.com/tehnik%D0%B0) item / 176-os-smartfony

11. Огляд платформи Eclipse [Електронний ресурс] Режим доступу: http://hightech.in. ua / comntent / art-eclipse-platform

12. Скільки днів в році люди витрачають на смартфони [Електронний ресурс] Режим доступу: http: /lit-news.club / 13588422 5. Visual Studio і

Xamarin [Електронний ресурс] Режим доступу: https : //msdn.microsoft.com/ru-ru/library/mt299001.aspx

13. Android Studio. Wikipedia [Електронний ресурс] Режим доступу: https://ru.wikipedia.org/wiki/Android Studio

ДОДАТОК А

```
namespace AppTableLayout
{
   [Activity(Label = "@string/app_name", Theme = "@style/AppTheme", MainLauncher = true)]
   public class MainActivity : AppCompatActivity
   {
      protected override void OnCreate(Bundle savedInstanceState)
      {
         base.OnCreate(savedInstanceState);
         Xamarin.Essentials.Platform.Init(this, savedInstanceState);
         // Set our view from the "main" layout resource
         SetContentView(Resource.Layout.activity_main);
         Init();
      }
      // Необходимые инициализации перед визуализацией приложения.
      private void Init()
      {
         // Получение текстовой информации из базы данных.
         Stream xmlDB = Assets.Open("stories.xml", Android.Content.Res.Access.Streaming);
```

```
// Для облегчения обработки базу данных загружаем в структурированный XML документ.
XmlDocument xmlDocument = new XmlDocument();
xmlDocument.Load(xmlDB);
xmlDB.Close();
// Получаем все названия рассказов.
XmlNodeList xmlNodeList = xmlDocument.DocumentElement.SelectNodes("story/name");
// В дизайнере кнопки этой радиогруппы только для визуального построения макета приложения.
// Очистим радиогруппу для размещения рабочих RadioButtons.
RadioGroup radioGroup = this.FindViewById<RadioGroup>(Resource.Id.radioGroupSelect);
radioGroup.RemoveAllViews();
// Создаем новые радиокнопки с названиями аналогично
// названиям элементов в файле XML.
// Подключаем событие прикосновения к каждой кнопке.
foreach (XmlNode item in xmlNodeList)
{
    RadioButton rb = new RadioButton(this);
    rb.Click += RadioGroupSelect_Click;
    rb.Text = item.InnerText;
```

```
            radioGroup.AddView(rb);
        }
        // Подключаем события для RadioButtons
        // цветовых настроек фона и шрифта книги.
        RadioGroup radioGroupBGColor =
this.FindViewById<RadioGroup>(Resource.Id.radioGroupBGColor);
        for (int i = 0; i < radioGroupBGColor.ChildCount; i++)
        {
            RadioButton rb = (RadioButton)radioGroupBGColor.GetChildAt(i);
            rb.Click += RadioButtonsSetting_Click;
        }



        // Подключаем события к группе радиокнопок изменения размера
шрифта книги.
        // У всех радиокнопок настроек события обрабатывает один и тот же
метод.
        RadioGroup radioGroupScaleFont =
this.FindViewById<RadioGroup>(Resource.Id.radioGroupScaleFont);
        for (int i = 0; i < radioGroupScaleFont.ChildCount; i++)
        {
```

```
            RadioButton rb = (RadioButton)radioGroupScaleFont.GetChildAt(i);
            rb.Click += RadioButtonsSetting_Click;
        }



        // Подключаем к текстовому полю событие долгого прикосновения.
        // При этом будет скрывается боковой столбец настроек.
        var textView =
this.FindViewById<TextView>(Resource.Id.textViewBody);
        textView.LongClick += TextView_LongClick;
    }



    // Метод событий нажатия(прикосновения) радиокнопок настроек.
    private void RadioButtonsSetting_Click(object sender, System.EventArgs e)
    {
        // Главную таблицу используем для общего фона всех визуальных
элементов.
        TableLayout tl =
this.FindViewById<TableLayout>(Resource.Id.tableLayoutMain);
```

```
        // Настраиваем шрифты заголовка и текстового поля.
        TextView textViewHead =
this.FindViewById<TextView>(Resource.Id.textViewHeader);
        TextView textView =
this.FindViewById<TextView>(Resource.Id.textViewBody);



        RadioButton rb = (RadioButton)sender;
        switch (rb.Id)
        {
           case Resource.Id.radioButtonDay:

              // Светлый фон и темный шрифт для дневного чтения книги.
              tl.SetBackgroundColor(Color.White);
              textViewHead.SetTextColor(Color.DarkGray);
              textView.SetTextColor(Color.DarkGray);

              break;
           case Resource.Id.radioButtonNight:
```

```
            // Темный фон и светлый шрифт для ночного чтения книги.
            tl.SetBackgroundColor(Color.DarkGray);
            textViewHead.SetTextColor(Color.LightGray);
            textView.SetTextColor(Color.LightGray);

            break;
        case Resource.Id.radioButtonFontNormal:

            // Стандартный шрифт.
            textView.TextSize = 14.0f;

            break;
        case Resource.Id.radioButtonFontLarge:

            // Увеличенный шрифт.
            textView.TextSize = 18.0f;

            break;
    }
}
```

```
        // Метод события долгого прикосновения к текстовому полю.
        private void TextView_LongClick(object sender, View.LongClickEventArgs
e)
        {
            // Получаем главную таблицу для управления столбцом настроек.
            TableLayout tl =
this.FindViewById<TableLayout>(Resource.Id.tableLayoutMain);



            // Если столбец свернут, возвращаем false для разворачивания,
            // и если столбец развернут возвращаем true для его сворачивания.
            bool collapse = tl.IsColumnCollapsed(1) ? false : true;



            // Выполняем вычисленное действие.
            tl.SetColumnCollapsed(1, collapse);
```

```
        }


        // Метод события клика радиокнопок группы выбора рассказа по его названию.
        private void RadioGroupSelect_Click(object sender, System.EventArgs e)
        {

            RadioButton rb = (RadioButton)sender;



            // Заголовок и тело текстового поля.
            TextView textViewHeader = this.FindViewById<TextView>(Resource.Id.textViewHeader);
            TextView textViewBody = this.FindViewById<TextView>(Resource.Id.textViewBody);



            // Получаем ресурс текстовой информации книги.
```

```
        Stream xmlDB = Assets.Open("stories.xml",
Android.Content.Res.Access.Streaming);



        // Для удобства работы текстовый ресурс загружаем в XML документ.
        XmlDocument xmlDocument = new XmlDocument();
        xmlDocument.Load(xmlDB);
        xmlDB.Close();



        // По тексту нажатой радиокнопки загружаем соответствующий
рассказ
        // и название присваиваем заголовку приложения.
        XmlNodeList xmlNodeList =
xmlDocument.DocumentElement.SelectNodes("story");
        foreach (XmlNode item in xmlNodeList)
        {
            if (rb.Text == item.SelectSingleNode("name").InnerText)
            {
                textViewBody.Text = item.SelectSingleNode("body").InnerText;
                textViewHeader.Text = item.SelectSingleNode("name").InnerText;
```

```
            }
        }
    }


    public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum] Android.Content.PM.Permission[] grantResults)
    {
        Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

        base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
    }
  }
}
using System.Reflection;
using System.Runtime.InteropServices;

// General Information about an assembly is controlled through the following
// set of attributes. Change these attribute values to modify the information
// associated with an assembly.
```

```
[assembly: AssemblyTitle("AppTableLayout")]

[assembly: AssemblyDescription("")]

[assembly: AssemblyConfiguration("")]

[assembly: AssemblyCompany("")]

[assembly: AssemblyProduct("AppTableLayout")]

[assembly: AssemblyCopyright("Copyright InterestPrograms.RU ©  2020")]

[assembly: AssemblyTrademark("")]

[assembly: AssemblyCulture("")]

[assembly: ComVisible(false)]
```

TableLayout

```
    xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    xmlns:p4="http://xamarin.com/mono/android/designer"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:background="#ffffff"

    android:id="@+id/tableLayoutMain">

    <TextView

        android:layout_width="match_parent"

        android:layout_height="wrap_content"
```

```
    android:text="Заголовок"
    android:textAlignment="center"
    android:textSize="22dp"
    android:id="@+id/textViewHeader" />
<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:id="@+id/tableRowContent">
    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:fillViewport="true"
        android:id="@+id/scrollViewBody" >
        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:text="Body"
            android:padding="10dp"
            android:textSize="14dp"
```

```
        android:id="@+id/textViewBody" />
</ScrollView>
<ScrollView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingRight="10dp">
<TableLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/tableLayout3">
    <RadioGroup
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:id="@+id/radioGroupSelect">
        <RadioButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="RadioButton"
            android:id="@+id/radioButton1" />
        <RadioButton
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RadioButton"
        android:id="@+id/radioButton2" />
    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RadioButton"
        android:id="@+id/radioButton3" />
</RadioGroup>
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:text="Настройки"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:id="@+id/textViewHeadSetting" />
<RadioGroup
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/radioGroupBGColor">
```

```
    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true"
        android:text="День"
        android:id="@+id/radioButtonDay" />
    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Ночь"
        android:id="@+id/radioButtonNight" />
</RadioGroup>
<RadioGroup
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp"
        android:id="@+id/radioGroupScaleFont">
    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Шрифт +"
```

```
            android:id="@+id/radioButtonFontLarge" />
        <RadioButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Шрифт -"
            android:id="@+id/radioButtonFontNormal" />
      </RadioGroup>
    </TableLayout>
    </ScrollView>
  </TableRow>
</TableLayout>
```